On the quality and complexity of Pareto equilibria in the Job Scheduling Game

Leah Epstein
Department of Mathematics
University of Haifa
31905 Haifa, Israel
lea@math.haifa.ac.il

Elena Kleiman
Department of Mathematics
University of Haifa
31905 Haifa, Israel
elena.kleiman@gmail.com

ABSTRACT

In the well-known scheduling game, a set of jobs controlled by selfish players wishes each to minimize the load of the machine on which it is executed, while the social goal is to minimize the makespan, that is, the maximum load of any machine. We consider this problem on the three most common machines models, identical machines, uniformly related machines and unrelated machines, with respect to both weak and strict Pareto optimal Nash equilibria. These are kinds of equilibria which are stable not only in the sense that no player can improve its cost by changing its strategy unilaterally, but in addition, there is no alternative choice of strategies for the entire set of players where no player increases its cost, and at least one player reduces its cost (in the case of strict Pareto optimality), or where all players reduce their costs (in the case of weak Pareto optimality).

We give a complete classification of the social quality of such solutions with respect to an optimal solution, that is, we find the Price of Anarchy of such schedules as a function of the number of machines, m. In addition, we give a full classification of the recognition complexity of such schedules.

Categories and Subject Descriptors

K.6.0 [**Management of Computing and information Systems**]: General—*Economics*; F.2.2 [**Nonnumerical Algorithms and Problems**]: [Sequencing and scheduling]

General Terms

Algorithms, Economics, Theory

Keywords

Economic paradigms: Economically-motivated agents, Game Theory (cooperative and non-cooperative), Price of Anarchy, Job Scheduling

1. INTRODUCTION

The rise of the Internet as a global platform for communication, computation, and commerce brought up the necessity to reconsider the prevalent paradigm in system design which assumes a central authority which constructs and manages the network and its participants, with a purpose of optimizing a global social objective.

Cite as: On the quality and complexity of Pareto equilibria in the Job Scheduling Game, Leah Epstein, Elena Kleiman, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. 525-532.

Designing a protocol intended for use in a global telecommunication network such as the Internet, we have to take into account that it consists of multiple independent and self-interested users, or players, which strive to optimize their private objective functions, also known as individual costs. In networks of such scale and complexity and in presence of raw economic competition between the parties involved, there is no possibility to introduce a single regulatory establishment enforcing binding commitments on the players.

Obviously, such collective behavior often leads to sub-optimal performance of the system, which is highly undesirable. In light of the above, there is an increased need to design efficient protocols that motivate self-interested agents to cooperate. Here "cooperation" may be defined as any enforceable commitment that makes it rational for the self interested players to choose a given strategic profile. In the settings in discussion, any meaningful agreement between the players must be self-enforcing. When deciding which particular strategy profile to offer for the users, the first and most basic requirement one has to consider is its stability, in a sense that no player would have an interest to unilaterally defect from this profile, given that the other players stick to it. This is consistent with the notion of Nash equilibrium (NE) [25], which is a widely accepted concept of stability in non-cooperative game theory. The second requirement is that the profile must be efficient. A fundamental concept of efficiency considered in economics is the Pareto efficiency, or Pareto optimality [24]. This efficiency criterion assures that it is not possible for a group of players to change their strategies so that every player is better off (or no worse off) than before.

One may justifully argue that Nash stability and Pareto optimality should be minimal requirements for any equilibrium concept intended to induce self-enforceability in presence of selfishness.

There are even stronger criteria for self-enforceability, requiring fairness in terms of fair competition without coalitions (like cartels and syndicates), and demanding from the profile to be resilient to groups (or coalitions) of players willing to coordinate their decisions, in order to achieve mutual beneficial outcomes. This is compatible with the definition of Strong Nash equilibrium (SNE) [3]. However, this requirement is sometimes too strong that it excludes many reasonable profiles.

We therefore restrict ourselves to profiles that satisfy the requirements of Nash stability and Pareto efficiency. In a sense, Pareto optimal Nash equilibria can be considered as intermediate concepts between Nash and Strong Nash equilibria; One may think of a Pareto optimal equilibrium as being stable under moves by single players or the grand coalition of all players, but not necessarily arbitrary coalitions. We distinguish between two types of Pareto efficiency. In a *weakly* Pareto optimal Nash equilibrium (WPO-NE) there is no alternative strategy profile beneficial for all players. A *strictly* Pareto optimal Nash equilibrium (SPO-NE) is also stable against deviations in which some players do not benefit but are also not worse off and at least one player improves his personal cost. Obviously, any *strictly* Pareto optimal equilibrium is also *weakly* Pareto optimal, but not wise-versa.

In this paper we consider strict and weak Pareto optimal Nash equilibria for scheduling games on the most common three machine models in the setting of pure strategies. This class of games is particularly important to our discussion as it models a great variety of problems in modern networks. Example applications include bandwidth sharing in ATM networks [7], market-based protocols for scheduling or task allocation [28], and congestion control protocols [18].

1.1 Model and Notation

We now define the general job scheduling problem. There are n jobs $J = \{1, 2, \ldots, n\}$ which are to be assigned to a set of m machines $M = \{M_1, \ldots, M_m\}$. We study three models of machines, that differ in the relation between the processing times of jobs on different machines. In the most general model of *unrelated* machines, job $1 \leq k \leq n$ has a processing time of p_{ik} on machine M_i, i.e., processing times are machine dependent. In the *uniformly related* (or *related*) machine model, each machine M_i for $1 \leq i \leq m$ has a speed s_i and each job $1 \leq k \leq n$ has a positive size p_k. The processing time of job k on machine M_i is then $p_{ik} = \frac{p_k}{s_i}$. If $p_{jk} = p_{j'k} = p_k$ for each job k and machines M_i and $M_{i'}$, the machines are called *identical* (in which case it is typically assumed the all speed are equal to 1).

An assignment or schedule is a function $\mathcal{A} : J \rightarrow M$. The load of machine M_i, which is also called the delay of this machine, is $L_i = \sum_{k:\mathcal{A}(k)=M_i} p_{ik}$. The cost, or the *social cost* of a schedule is the maximum delay of any machine, also known as the *makespan*, which we would like to minimize.

The job scheduling game JS is characterized by a tuple $JS = \langle N, (\mathcal{M}_k)_{k\in N}, (c_k)_{k\in N}\rangle$, where N is the set of atomic players. Each selfish player $k \in N$ controls a single job and selects the machine to which it will be assigned. We associate each player with the job it wishes to run, that is, $N = J$. The set of strategies $\mathcal{M}_k$ for each job $k \in N$ is the set M of all machines. i.e. $\mathcal{M}_k = M$. Each job must be assigned to one machine only. Preemption is not allowed. The outcome of the game is an assignment $\mathcal{A} = (\mathcal{A}_k)_{k\in N} \in \times_{k\in N} M_k$ of jobs to the machines, where $\mathcal{A}_k$ for each $1 \leq k \leq n$ is the index of the machine that job k chooses to run on. Let $\mathcal{S}$ denote the set of all possible assignments.

The cost function of job $k \in N$ is denoted by $c_k : \mathcal{S} \rightarrow \mathbb{R}$. The cost c_k^i charged from job k for running on machine M_i in a given assignment $\mathcal{A}$ is defined to be the load observed by machine i in this assignment, that is $c_k(i, \mathcal{A}_{-k}) = L_i(\mathcal{A})$, when $\mathcal{A}_{-k} \in \mathcal{S}_{-k}$; here $\mathcal{S}_{-k} = \times_{j\in N\setminus\{k\}}\mathcal{S}_j$ denotes the actions of all players except for player k. The goal of the selfish jobs is to run on a machine with a load which is as small as possible. Similarly, for $K \subseteq N$ we denote by $\mathcal{A}_K \in \mathcal{S}_{-K}$ the set of strategies of players outside of K in a strategy profile $\mathcal{A}$, when $\mathcal{S}_{-K} = \times_{j\in N\setminus K}\mathcal{S}_j$ is the action space of all players except for players in K. The social cost of a strategy profile $\mathcal{A}$ is denoted by $SC(\mathcal{A}) = \max\limits_{1\leq k\leq n} c_k(\mathcal{A})$.

We will next provide formal definitions of Nash, Weak/Strict Pareto Nash and Strong Nash equilibria in the job scheduling game, using the notations given above.

DEFINITION 1. *(Nash equilibrium) A strategy profile $\mathcal{A}$ is a (pure) Nash equilibrium (*NE*) in the job scheduling game JS if for all $k \in N$ and for any strategy $\bar{\mathcal{A}}_k \in M$, $c_k(\mathcal{A}_k, \mathcal{A}_{-k}) \leq c_k(\bar{\mathcal{A}}_k, \mathcal{A}_{-k})$.*

It was shown that job scheduling games always have (at least one) pure Nash equilibrium [15, 11]. We denote the set of Nash equilibria of an instance G of the job scheduling game by NE(G).

DEFINITION 2. *(Strong Nash equilibrium) A strategy profile $\mathcal{A}$ is a Strong Nash equilibrium (*SNE*) in the job scheduling game JS if for every coalition $\phi \neq K \subseteq N$ and for any set of strategies $\bar{\mathcal{A}}_K \in \times_{j\in K}\mathcal{M}_j$ of players in K, there is a player $i \in K$ such that $c_i(\bar{\mathcal{A}}_K, \mathcal{A}_{-K}) \geq c_i(\mathcal{A}_K, \mathcal{A}_{-K})$.*

Existence of Strong Nash equilibrium in job scheduling games was proved in [1]. We denote the set of Strong Nash equilibria of an instance G of the job scheduling game by SNE(G).

Clearly SNE(G)⊆NE(G), as coalitions of size 1 can not improve by changing their strategy.

DEFINITION 3. *(Weak/Strict Pareto optimal profile) A strategy profile $\mathcal{A}$ is weakly Pareto optimal (WPO) if there is no strategy profile $\bar{\mathcal{A}}$ s.t. for all $k \in N$, $c_k(\bar{\mathcal{A}}) < c_k(\mathcal{A})$.*

A strategy profile $\mathcal{A}$ is strictly Pareto optimal (SPO) if there is no strategy profile $\bar{\mathcal{A}}$ and $k^* \in N$ s.t. for all $k \in N\backslash k^*$, $c_k(\bar{\mathcal{A}}) < c_k(\mathcal{A})$ and $c_{k^*}(\bar{\mathcal{A}}) \leq c_{k^*}(\mathcal{A})$.

We denote by SPO(G) and WPO(G), respectively, the sets of strictly and weakly Pareto optimal profiles of an instance G of the job scheduling game. Clearly, SPO(G)⊆ WPO(G).

A strategy profile $\mathcal{A} \in$ NE(G)∩ WPO(G) is called Weak Pareto optimal Nash equilibrium (WPO-NE), and a strategy profile $\mathcal{A} \in$NE(G) ∩ SPO(G) is called Strict Pareto optimal Nash equilibrium (SPO-NE), and these are the profiles that we focus on.

We note that every strong equilibrium is also weakly Pareto optimal, as the requirement in Definition 2 applies to the grand coalition of all players. Hence SNE(G)⊆ WPO(G). The existence of Strong Nash equilibria in job scheduling games assures the existence of weak Pareto optimal Nash equilibria.

On the other hand, in general, neither Nash equilibria nor Strong Nash equilibria are necessarily strictly Pareto optimal. Existence of strict Pareto optimal Nash equilibria in scheduling games (among others) was proved in [16].

An important issue concerns the quality of these solution concepts. As there is a discrepancy between the private goals of the players and the global social goal, we would like to measure the loss in the performance of the system as it is reflected by the closeness of the costs of these concepts to the cost of the optimal solution, when the accepted methodology is worst-case approach.

The quality measures which consider Nash equilibria are the Price of Anarchy introduced by Koutsoupias and Papadimitriou [20] and the more optimistic Price of Stability suggested by Anshelevich et al. [2], which are defined as the worst-case ratio between the social cost of the worst/best Nash equilibrium to the social cost of an optimal solution, which is denoted by OPT. Formally,

DEFINITION 4. *(Price of Anarchy and Stability) The Price of Anarchy (PoA) of the job scheduling game JS is defined by*

$$PoA(JS) = \sup_{G\in JS} \sup_{\mathcal{A}\in NE(G)} \frac{SC(\mathcal{A})}{\text{OPT}(G)}.$$

If instead we consider the best Nash equilibrium of every instance, this leads to the definition of the Price of Stability (PoS):

$$PoS(JS) = \sup_{G\in JS} \inf_{\mathcal{A}\in NE(G)} \frac{SC(\mathcal{A})}{\text{OPT}(G)}.$$

This concept is applied analogously to Strong Nash equilibria as well as to weakly/strictly Pareto optimal Nash equilibria yielding the Strong Price of Anarchy $SPoA(JS)$ and the Strong Price of Stability $SPoS(JS)$ as well as the weak and strict Pareto Prices of Anarchy $WPO\text{-}PoA(JS)$, $SPO\text{-}PoA(JS)$ and Stability $WPO\text{-}PoS(JS)$, $SPO\text{-}PoS(JS)$. By definition, it is clear that $SPoA(JS) \leq WPO\text{-}PoA(JS) \leq PoA(JS)$. As any *strictly* Pareto optimal NE is also a *weakly* Pareto optimal NE, it must be the case that $WPO\text{-}PoA(JS) \geq SPO\text{-}PoA(JS)$. However, we can show that in the job scheduling game there is no immediate relation between the $SPO\text{-}PoA(JS)$ and the $SPoA(JS)$, as there are Strong Nash equilibria that are not *strictly* Pareto optimal, while there are *strictly* Pareto optimal Nash equilibria that are not strong equilibria.

Some natural questions in this context are whether the Pareto Prices of Anarchy are significantly smaller than the standard Price of Anarchy, whether the weak Pareto Price of Anarchy is much larger than the Strong Price of Anarchy, and finally, whether there is any relation between the Strong Price of Anarchy and the strict Pareto Price of Anarchy. In other words, does the requirement that the equilibrium must be Pareto optimal leads to greater efficiency, and does the further demand that the equilibrium must be stable against arbitrary coalitions is helpful.

1.2 Related work and our contribution

Pareto efficiency of resource assignments is a well referred issue in economics, especially in welfare economics. Pareto efficiency is a highly desirable trait, however Dubey [9] has shown that Nash equilibria may generally be Pareto inefficient based on the difference between the conditions to be satisfied by Nash equilibria and those to be satisfied by Pareto optima.

Job scheduling is a classical problem in combinatorial optimization. The analysis of job scheduling in the algorithmic game theory context was initiated by Koutsoupias and Papadimitriou in their seminal work [20], which was followed by many others (see e.g. [8, 22, 1, 13]). In our overview of the known results we will limit our discussion only to results concerning pure strategies. We will begin with the results on quality measures that concern Nash equilibria of the game. For m identical machines, the PoA is $2 - \frac{2}{m+1}$ which can be deduced from the results of [14] (the upper bound) and [26] (the lower bound). For related machines the PoA is $\Theta(\frac{\log m}{\log \log m})$ [19, 8, 20]. In the model of unrelated machines the PoA is unbounded [5], which holds already for two machines. From the results of [11] it is evident that in all three models the PoS is 1. The study of quality measures that concern Strong Nash equilibria of this game was initiated by Andelman at el. [1]. For identical machines, they proved that the $SPoA$ equals the PoA, which in turn equals $2 - \frac{2}{m+1}$. For related machines, Fiat et al. [1] showed that the $SPoA$ is $\Theta(\frac{\log m}{(\log \log m)^2})$. Surprisingly, the $SPoA$ for this problem is bounded by the number of machines m, as shown in [13], and this is tight [1]. Andelman at el. also showed that $SPoS$ is 1.

The previous work on Pareto efficiency of Nash equilibria in algorithmic game theory was mainly concerned with weak Pareto equilibria, probably since a solution which is not weakly Pareto optimal is clearly unstable. A textbook in economics states the following: "The concept of Pareto optimality originated in the economics equilibrium and welfare theories at the beginning of the past century. The main idea of this concept is that society is enjoying a maximum ophelimity when no one can be made better off without making someone else worse off" [21]. Thus the strict Pareto is a stronger and more meaningful efficiency notion, as it captures an important aspect of human social behavior. Another issue is that the weak Pareto implies that everyone prefers some assignment to any other. In reality, such unanimity of preferences among all persons is very rare. To conclude, both concepts are important, and we focus on both of them in this work.

Pareto optimality of Nash equilibria has been studied in the context of congestion games, see Chien and Sinclair [6] and Holzman and Law-Yone [17]. The former gave conditions for uniqueness and for weak and strict Pareto optimality of Nash equilibria, and the latter characterized the weak Pareto Prices of Anarchy and Stability. The existence, and complexity of recognition and computation of weak Pareto Nash equilibria in congestion games was studied recently by Hoefer and Skopalik in [27].

In [16] Harks at el. show that a class of games that have a *Lexicographical Improvement Property* (which our game indeed has) admits a generalized strong ordinal potential function. They use this to show existence of Strong Nash equilibria with certain efficiency and fairness properties in these games, strict Pareto efficiency included. They do so by arguing that a player wise cost-lexicographically minimal assignment is also strictly Pareto optimal (and so it is optimal w.r.t. the social goal function as well).

Weak Pareto Nash equilibria in routing and job scheduling games were considered recently in [4] by Aumann and Dombb. As a measure for quantifying the distance of a best/worst Nash equilibrium from being weakly Pareto efficient, they use the smallest factor by which any player improves its cost when we move to a different strategy profile, which they refer to as "Pareto inefficiency". They do not consider however the quality of Pareto optimal Nash equilibria with respect to the social goal.

Among other results, it is shown in [4] that any Nash equilibrium assignment is necessarily weakly Pareto optimal for both identical and related machines. Moreover, for any machine model, any assignment which achieves the social optimum must be weakly Pareto optimal. One such assignment is one whose sorted vector of machine loads is lexicographically minimal is necessarily weakly Pareto optimal (see also [1, 11]). Milchtaich [23] has proved related results for the case of non-atomic players, where the processing time of each player is negligible compared to the total processing time.

We consider these issues for SPO-NE assignments. We show that while the property of identical machines remains true, this is not the case for related machines, that is, not every Nash equilibrium assignment is strict Pareto optimal. For unrelated machines, while there always exist an assignment which is a social optimum and a SPO-NE, assignments with lexicographically minimal sorted vector of machine loads are not necessarily strictly Pareto optimal. In this paper we fully characterize the weak and strict Pareto Prices of Anarchy of the job scheduling game in cases of identical, related and unrelated machines. The characterization of the Prices of Stability follows from previous work as explained above.

Next, we consider the complexity of recognition of weak and strict Pareto optimality of NE. Note that the recognition of NE can be done in polynomial time for any machine model by examining potential deviations of each job. As for strong equilibria, it was shown by Feldman and Tamir [12] that it is NP-hard to recognize an SNE for $m \geq 3$ identical machines and for $m \geq 2$ unrelated machines. For two identical machines, they showed that any NE is a SNE, so recognition can be done in polynomial time (for $m \geq 3$, it was shown in [1] that not every NE is a SNE). For the only remaining case of two related machines, it was shown [10] that recognition is again NP-hard. We show that the situation for Pareto optimal equilibria is slightly different. In fact, recognition of WPO-NE or SPO-NE can be done in polynomial time for identical machines and related machines. For unrelated machines, we show that the recognition of WPO-NE is NP-hard in the strong sense and the recognition of SPO-NE is NP-hard.

We reflect upon the differences between the results for weak and strict Pareto equilibria also compared to strong equilibria, and make conclusions regarding the relations between the quality measures in this game. See Table 1 for a summary of the results.

2. PARETO PRICES OF ANARCHY IN THE JOB SCHEDULING GAME

2.1 Identical and Related machines

A result from [4] shows that any NE schedule for identical and related machines is weakly Pareto optimal. This result implies that $WPO\text{-}PoA(JS) = PoA(JS)$. For the case of identical machines, they give an even stronger result: every schedule where every machine receives at least one job is weakly Pareto optimal. Note that if $n < m$, then a schedule is weakly Pareto optimal if and only if at least one machine has a single job (to obtain strict Pareto optimality for this case, or to obtain a NE, each job needs to be assigned to a different machine).

In the strict Pareto case, while the general result for identical machines still holds, and the set of NE schedules is equal to the set of SPO-NE schedules for identical machines (as we prove next), it is not necessarily true for related machines. We exhibit an example of a schedule which is a NE but it is not strictly Pareto optimal.

Consider a job scheduling game with two related machines of speeds 1,2 and two jobs of size 2. There are two types of pure NE schedules: in the first one, both jobs are assigned to the fast machine, and in the other one job runs on each machine. The first one is not a SPO-NE, as switching to a schedule of the second type strictly reduces the cost for one of the jobs, while not harming the other. Moreover, the sorted machine load vector of the first type of schedules is $(2, 0)$, while the load vector of the second type is $(2, 1)$, so the schedule with the lexicographically minimal machine load vector is not a SPO-NE (even though it is a SNE).

This difference in the results for related machines is explained by the fact that conditions for weak Pareto allow Pareto improvements where not all jobs strictly improve while the strict Pareto does not. If a NE schedule has an empty machine, and a job arrived to such a machine as a result of a deviation to a different schedule, where all jobs strictly reduce their costs, then the reduction in the cost of this job contradicts the original schedule being a NE. However, if the job only needs to maintain its previous cost, then there is no contradiction.

We will prove the following theorem, extending the result of [4] which will allow us to claim that for identical machines, $SPO\text{-}PoA(JS) = PoA(JS)$.

THEOREM 5. *Any schedule for identical machines, where no machine is empty, is strictly Pareto optimal.*

This is a stronger result than the one in [4], since it deals with strict Pareto. The idea of the proof goes along the lines of [4], but we need to modify it so that it applies for the stronger conditions of strict Pareto optimal schedules. First we prove the following property, which we will also use to characterize the $WPO\text{-}PoA$ for related machines.

THEOREM 6. *Consider a schedule X that is not a SPO-NE, and denote the set of non-empty machines (which receive at least one job) in X by μ_X. Let Y be a different schedule where no job has a larger cost than it has in X and at least one job has a smaller cost. Denote the set of non-empty machines in Y by μ_Y. Then,*

$$\sum_{i\in\mu_X} s_i < \sum_{i\in\mu_Y} s_i,$$

where s_i is the speed of machine i.

PROOF. Consider a transition from schedule X to schedule Y, and denote by x_i^j the sum of the sizes of jobs that are moved from machine $i \in \mu_X$ to machine $j \in \mu_Y$ (for $j = i$, this gives the sum of sizes of jobs that are assigned to this machine in both schedules). Let ℓ_t, for $t \in \mu_X$, be the sum of sizes of jobs that run on machine t in X, and let ℓ'_t, for $t \in \mu_Y$, be the sum of sizes of jobs that run on machine t in Y. We extended the definition so that if $t \notin \mu_X$, then $\ell_t = 0$, and if $t \notin \mu_Y$, then $\ell'_t = 0$.

Consider the total sum of sizes of jobs assigned to a machine in X or in Y, then the following claim holds:

CLAIM 7. *For every $i \in \mu_X$, $\sum\limits_{j\in\mu_Y} x_i^j = \ell_i$, or $\sum\limits_{j\in\mu_Y} \frac{x_i^j}{\ell_i} = 1$. For every $j \in \mu_Y$, $\sum\limits_{i\in\mu_X} x_i^j = \ell'_j$, or $\sum\limits_{i\in\mu_X} \frac{x_i^j}{\ell'_j} = 1$.*

By the definition of the costs in Y compared to X, we get that:

CLAIM 8. *If $x_i^j > 0$, then $\frac{\ell'_j}{s_j} \leq \frac{\ell_i}{s_i}$, and there exist $i \in \mu_X$,$i \in \mu_Y$ such that $\frac{\ell'_j}{s_j} < \frac{\ell_i}{s_i}$.*

The following also holds:

CLAIM 9. *For every $i \in \mu_X$, $j \in \mu_Y$: $\frac{x_i^j}{\ell_i} \leq \frac{s_j}{s_i} \cdot \frac{x_i^j}{\ell'_j}$, and there exist i, j such that $\frac{x_i^j}{\ell_i} < \frac{s_j}{s_i} \cdot \frac{x_i^j}{\ell'_j}$.*

PROOF. As $i \in \mu_X$, $\ell_i > 0$, as $j \in \mu_Y$, $\ell'_i > 0$. If $x_i^j > 0$, it is derived from Claim 8, if $x_i^j = 0$ it holds trivially. Since there is at least one job for which the cost in Y is strictly smaller than its cost in X, then the second property must hold. □

Summing up the inequalities in Claim 9 over all $j \in \mu_Y$, in combination with Claim 7, we get that for any $i \in \mu_X$:

$1 = \sum_{j\in\mu_Y} \frac{x_i^j}{\ell_i} \leq \sum_{j\in\mu_Y} \frac{s_j}{s_i} \cdot \frac{x_i^j}{\ell'_j}$, where there is at least one $i \in \mu_X$ for which this inequality is strict. Equivalently, $s_i \leq \sum_{j\in\mu_Y} \frac{x_i^j \cdot s_j}{\ell'_j}$. Summing up the last inequality over all $i \in \mu_X$ combined with the fact that for some i this inequality is strict, changing the order of summation, and using Claim 7 we get: $\sum_{i\in\mu_X} s_i < \sum_{i\in\mu_X}\sum_{j\in\mu_Y} \frac{x_i^j \cdot s_j}{\ell'_j} = \sum_{j\in\mu_Y}\sum_{i\in\mu_X} \frac{x_i^j \cdot s_j}{\ell'_j} = \sum_{j\in\mu_Y} s_j$, which concludes our proof.

We now return to the proof of Theorem 5.

PROOF. We show that any schedule X for identical machines where μ_X=M is strictly Pareto optimal. Assume by contradiction that this is not the case, and hence there exists a different schedule Y where at least one job improves, while all the other jobs are not worse off. As the machines in question are identical, $s_1 = s_1 = \ldots = s_m$ holds, thus $\sum_{i\in\mu_X} s_i = m$ and $\sum_{j\in\mu_Y} s_j \leq m$. By Theorem 6 we get that $m = \sum_{i\in\mu_X} s_i < \sum_{j\in\mu_Y} s_j \leq m$, which is a contradiction, and we conclude that such Y cannot exist. □

COROLLARY 10. *Every schedule on identical machines which is a NE is also a SPO-NE. Thus in this case $SPO\text{-}PoA$=$WPO\text{-}PoA$=PoA.*

PROOF. Consider a NE schedule. If there is an empty machine, then each machine has at most one job (otherwise, if some machine has two jobs then any of them can reduce its cost by moving to an empty machine), and thus each job has the smallest cost that it can have in any schedule. Otherwise, the property follows from Theorem 5. □

	# of machines	Strict Pareto			Weak Pareto		
		SPO-PoA	SPO-PoS	Recognition	WPO-PoA	WPO-PoS	Recognition
identical	m	$2-\frac{2}{m+1}$	$\mathbf{1}$ [16]	P	$2-\frac{2}{m+1}$	$\mathbf{1}$ [1, 4]	P
related	m	$\Theta(\frac{\log m}{\log\log m})$	$\mathbf{1}$ [16]	P	$\Theta(\frac{\log m}{\log\log m})$	$\mathbf{1}$ [1, 4]	P
unrelated	$m=2$	2	$\mathbf{1}$ [16]	NP-hard	2	$\mathbf{1}$ [1, 4]	NP-hard
unrelated	$m\geq 3$	m	$\mathbf{1}$ [16]	NP-hard	∞	$\mathbf{1}$ [1, 4]	NP-hard

Table 1: Summary of Results

We next consider related machines and prove that the three measures are equal in this case as well.

THEOREM 11. *In the job scheduling game on related machines $SPO\text{-}PoA$=$WPO\text{-}PoA$=PoA.*

PROOF. As any SPO-NE is also a WPO-NE, and every WPO-NE is a NE, the following sequence of inequalities holds: $SPO\text{-}PoA \leq WPO\text{-}PoA \leq PoA$. We will prove that this is actually a sequence of equalities. It is enough to prove that $PoA \leq SPO\text{-}PoA$. We will do it by showing that the lower bound example for the PoA given in [8] is also a lower bound for the $SPO\text{-}PoA$, by proving that it is strictly Pareto optimal.

For completeness, we first present the lower bound of [8]. Consider a job scheduling game on m related machines. The machines are partitioned into $k+1$ groups, each group j ,$0 \leq j \leq k$ has N_j machines. The sizes of the groups are defined in inductive manner: $N_k = \Theta(\sqrt{m})$, and for every $j < k$: $N_j = (j+1)\cdot N_{j+1}$ (and thus $N_0 = k!\cdot N_k$). The total number of machines $m = \sum_{j=0}^{k} N_j = \sum_{j=0}^{k} \frac{k!}{(k-j)!}\cdot N_k$. It follows that $k \sim \frac{\log m}{\log\log m}$. The speed of each machine in group j is $s_j = 2^j$.

A schedule is defined as follows: each machine in group j has j jobs, each with size 2^j. Each such job contributes 1 to the load of its machine. The load of each machine in group N_j is then j, and therefore the makespan which is accepted on the machines in group N_k is k. Note that all the machines in group N_0 are empty.

We denote this schedule by X. It was proven in [8] that X is a pure NE. We claim that it is also strictly Pareto optimal.

CLAIM 12. *X is strictly Pareto optimal.*

PROOF. Assume by contradiction that X is not a SPO-NE, so there exists another schedule Y where at least one job improves, and all the other jobs are not worse off. Observe that all the machines in group N_0 necessarily remain empty in Y; each job that runs on a machine in group N_j for $1 \leq j \leq k$ pays a cost of j in X, and if it is assigned on a machine from group N_0 in Y it has to pay a cost of 2^j, and $2^j > j$ for $j \geq 1$, which makes it strictly worse off. This means that $\mu_Y \subseteq \mu_X$. On the other hand, according to Theorem 6 which we proved earlier, $\sum\limits_{i\in\mu_X} s_i < \sum\limits_{i\in\mu_Y} s_i$ must hold, and we get a contradiction. Hence, the schedule in this example is strictly Pareto optimal. □

An optimal schedule has a makespan of 2. To obtain such a schedule, we move all jobs from machines in N_j ($j\cdot N_j$ jobs, each of size 2^j) to machines in N_{j-1}, for $1 \leq j \leq k$. Every machine gets at most one job, and the load on all machines is less or equal to $\frac{2^j}{2^{j-1}} = 2$. The $SPO\text{-}PoA$ is therefore $\Omega(\frac{\log m}{\log\log m})$.

We conclude that $SPO\text{-}PoA$=$WPO\text{-}PoA$=PoA.

It was proved in [13] that schedule X is not a SNE, as a coalition of all k jobs from a machine in group N_k with 3 jobs from each of k different machines from group N_{k-2} can jointly move in a way that reduces the costs of all its members. In addition to determining the SPO-NE, this example illustrates the point that in the job scheduling game not every SPO-NE is necessarily a SNE. We saw that for related machines, the $SPO\text{-}PoA$=$WPO\text{-}PoA$ are the same as the PoA, while the $SPoA$ is lower.

2.2 Unrelated machines

We saw that already for related machines, not every SNE is a SPO-NE and vice versa. However, the results which we find for the $SPO\text{-}PoA$ on unrelated machines are similar to those which are known for the $SpoA$, that is, the $SPO\text{-}PoA$ is equal to m for any number of machines m. Interestingly, the $WPO\text{-}PoA$ for the setting $m = 2$ is exactly 2, like the $SPO\text{-}PoA$, but for $m \geq 3$ it is unbounded like the PoA.

THEOREM 13. *There exists a job scheduling game with 2 unrelated machines, such that $WPO\text{-}PoA \geq 2$. For any, $m \geq 3$ there exists a job scheduling game with m unrelated machines, such that $WPO\text{-}PoA$ is unbounded.*

PROOF. Consider a job scheduling game with two unrelated machines and two jobs, where $p_{11} = p_{21} = p_{12} = 1$ and $p_{22} = 2$. A schedule where job 1 is assigned to M_1 and job 2 is assigned to M_2 with a makespan of 2 is a WPO-NE; No job would benefit from moving to a different machine, and job 1 will not profit by switching to a different schedule. In an optimal schedule for this game, job k, $k \in \{1,2\}$ is assigned to M_k, and the makespan is 1. We get that $WPO\text{-}PoA \geq 2$.

Now, consider a job scheduling game with $m \geq 3$ unrelated machines and $n = m$ jobs, where for each job k, $1 \leq k \leq m$: $p_{kk} = \varepsilon$, and $p_{jk} = 1$ for all $j \neq k$, for some arbitrary small positive ε. A schedule where job 1 is assigned to run on M_1, job m is assigned to M_2 and each job k for $2 \leq k \leq m-1$ is assigned to M_{k+1}, is a WPO-NE. It is weakly Pareto optimal since job 1 cannot decrease its cost by changing to any other assignment. The only way that another job could decrease its cost would be by moving to the machine where its cost is ε, but the load on all those machines is 1. Therefore, the schedule is a NE. The makespan of this schedule is 1. An optimal schedule for this game, where each job $1 \leq k \leq m$ is assigned to machine M_k, has a makespan of ε. In total, we have $WPO\text{-}PoA \geq \frac{1}{\varepsilon}$, which is unbounded letting ε tend to zero. □

We next prove a matching upper bound for $m = 2$.

THEOREM 14. *For any job scheduling game with 2 unrelated machines, $WPO\text{-}PoA \leq 2$.*

PROOF. Consider a schedule on two unrelated machines which is a WPO-NE. Without loss of generality, assume that the load of M_1 is not larger than the load of M_2, and denote the loads of the machines are by L_1 and L_2, respectively. The makespan of this schedule is then L_2. We show $L_2 \leq 2\text{OPT}$. We first show $L_1 \leq$

OPT. If $L_2 \geq L_1 >$ OPT then an optimal schedule has the property that every job has a smaller cost in it than it has in the current schedule (a cost of at most OPT $< L_1 \leq L_2$), in contradiction to the fact that this schedule is a WPO-NE.

To complete the proof, we upper bound L_2. If $L_2 \leq$ OPT, then we are done, otherwise, $L_2 >$ OPT, and there must exist a job k assigned to M_2 which is assigned to M_1 in an optimal schedule (since the load resulting from jobs assigned to M_2 in an optimal schedule is no larger than OPT). Thus, $p_{1k} \leq$ OPT, and in the alternative schedule, where this job moves to M_1, the new load of M_1 is at most $L_1 + p_{1k} \leq 2$OPT. However, we know that the given schedule is a NE, which means that $L_1 + p_{1k} \geq L_2$, giving $L_2 \leq 2OPT$. Therefore, $WPO\text{-}PoA \leq 2$. □

From Theorems 13 and 14 we conclude that for $m = 2$, $WPO\text{-}PoA = 2$, and for $m \geq 3$, $WPO\text{-}PoA = \infty$.

We prove next that like the $SPoA$, the $SPO\text{-}PoA$ is m. We should note that the previous results for the $SPoA$ cannot be used here. As we saw, the sets of SNE and SPO-NE have no particular relation. The proofs used for the $SPoA$ do not hold for the $SPO\text{-}PoA$ and need to be adapted. The lower bound of m on the $SPoA$ by Andelman et al. [1] is not strictly Pareto optimal (see below), and in the proof of the upper bound by Fiat et al. [13] the claim is proved by considering alternative schedules where the jobs which change their strategies are proper subsets of jobs (so other jobs may increase their costs).

THEOREM 15. *The $SPO\text{-}PoA$ for m unrelated machines in any job scheduling game is at most m.*

PROOF. Consider a schedule $\mathcal{A}$ on m unrelated machines which is a SPO-NE. Assume that the machines are sorted by non-increasing order of loads, that is, $L_1 \geq L_2 \geq \ldots \geq L_m$. The makespan of $\mathcal{A}$ is therefore L_1.

First, note that $L_m \leq OPT$. If $L_m > OPT$ then an optimal schedule has the property that every job has a smaller cost in it, contradicting the strict Pareto optimality of $\mathcal{A}$. Next, we will prove that $L_i - L_{i+1} \leq OPT$ holds for any $1 \leq i \leq m-1$. Assume by contradiction that there exists i so that $L_i - L_{i+1} > OPT$. We let $L_{i+1} = \delta$. By our assumption $L_i > \delta + OPT$ holds.

Now, consider another schedule $\mathcal{A}'$, where each one of the jobs from machines M_j for $1 \leq j \leq i$ in $\mathcal{A}$ is running on the machine on which it runs in an optimal schedule (all the other jobs hold their positions). We observe that none of these jobs runs on machines $M_{i+1}, \ldots, M_m$ in $\mathcal{A}'$ (or in the optimal schedule under consideration); The processing time of each such job in $\mathcal{A}'$ is at most OPT, and as $L_k \leq \delta$ for $i+1 \leq k \leq m$, its cost in $\mathcal{A}$ if it switches to the machines out of $M_{i+1}, \ldots, M_m$ on which its processing time is at most OPT, then the load of this machine would be at most $\delta + OPT$, while its cost in $\mathcal{A}$ was strictly larger than $\delta + OPT$, contradicting $\mathcal{A}$ being a NE.

We conclude that these jobs are scheduled in $\mathcal{A}'$ on machines $M_1, \ldots, M_i$, where the load of each one of the machines is at most OPT, and that the loads and the allocations on machines $M_{i+1}, \ldots,$ M_m do not change from $\mathcal{A}$ to $\mathcal{A}'$.

This means that in $\mathcal{A}'$ the costs of all jobs from machines $M_1, \ldots$ $, M_i$ in $\mathcal{A}$ are strictly improved, and the costs of all jobs from machines $M_{i+1}, \ldots, M_m$ in $\mathcal{A}$ do not change, which contradicts $\mathcal{A}$ being a SPO-NE. Hence, such i does not exist. Applying this inequality repeatedly, we get that $L_1 \leq L_m + (m-1)$OPT, which in combination with the fact that $L_m \leq OPT$ gives us $SPO\text{-}PoA \leq m$. □

THEOREM 16. *There exists an instance of job scheduling game with m unrelated machines for which $SPO\text{-}PoA \geq m$.*

PROOF. Consider a job scheduling game with m unrelated machines and $n = m$ jobs, where for each job k, $2 \leq k \leq m$: $p_{kk} = k - k\varepsilon$, $p_{k(k-1)} = 1$ and $p_{ik} = \infty$ for all $i \neq k-1, k$. For job 1, $p_{11} = 1 - \varepsilon$ (for some small positive $\varepsilon < \frac{1}{m}$), $p_{m1} = 1$ and $p_{i1} = \infty$ for all $i \neq 1, m$.

In an optimal schedule for this game each one of the jobs $2 \leq k \leq m$ runs alone on machine M_{k-1} and job 1 runs on M_m, which yields a makespan of 1.

On the other hand, a schedule where each one of the jobs $1 \leq k \leq m$ runs alone on machine M_k has a makespan of $m - m\varepsilon$. We will show that this schedule is a SPO-NE. The schedule is a NE, since for each job, moving to the only additional machine on which its processing time is not infinite increases it cost by at least ε. Consider an alternative schedule where no job increases its cost. Job 1 is currently assigned on a machine with load $1 - \varepsilon$, which is the minimal possible cost for it, and this minimum is unique. Thus any alternative schedule must keep job 1 assigned alone to the first machine. We can prove by induction on the indices of jobs that every job has to stay assigned to its current machine alone; once job k must stay on its machine alone, job $k+1$ does not have an alternative machine, and adding another job to the machine that it is assigned to (M_k) would increase its cost. Thus such a schedule does not exist. This gives that $SPO\text{-}PoA \geq m$. □

We conclude that for for any m, $SPO\text{-}PoA = m$.

This is a proper place to mention that the lower bound example from [1] showing that $SPoA \geq m$ looks similar to our example at a first glance. The difference in processing times is in the definition $p_{kk} = k$, for $1 \leq k \leq m$. However, this example does not apply here, as the schedule of cost m which it gives is not strictly Pareto optimal; if we switch to the optimal schedule, where job 1 runs on M_m and each job $2 \leq k \leq m$ runs on M_{k-1}, all jobs $2 \leq k \leq m$ strictly improve their costs and job 1 is not worse off.

This is another example which demonstrates the fact that in the job scheduling game we consider not every SNE is necessarily a SPO-NE. However, we showed that this is the case already for related machines.

3. RECOGNITION OF PARETO OPTIMAL EQUILIBRIA

In this section we consider the computational complexity of SPO-NE and WPO-NE for all machine models. Specifically, we investigate the problem of recognition of such schedules.

THEOREM 17. *There exists a polynomial time algorithms which receives a schedule on related machines (or on identical machines) and check whether the schedule is a* SPO-NE *and whether it is a* WPO-NE.

PROOF. Consider a schedule $\mathcal{A}$, and recall that one can determine in polynomial time whether a given schedule is a NE. Since any NE on identical machines is a WPO-NE and a SPO-NE, the recognition of such schedules is equivalent to recognition of NE. This is also the case for related machines and WPO-NE.

For the recognition of SPO-NE on related machines, we use the following algorithm. First, check whether the schedule is a NE (if not, then output a negative answer). If the schedule is a NE and it does not contain an empty machine, return a positive answer. Otherwise, for every job k, such that k is assigned to a machine which has at least two jobs assigned to it, test if moving it to an empty machine of maximum speed does not increase its cost. If there exists a job for which the cost is not increased, return a negative answer, and otherwise, a positive answer. Note that if there exists an empty machine, but no machine has two jobs assigned to it, then the returned answer is positive.

Now we prove correctness of the last algorithm. If there are no empty machines then any NE is a SPO-NE (by Theorem 6). For the remaining cases of the algorithm, we prove the following claim.

CLAIM 18. *Given $\mathcal{A}$, which is a* NE, *there exists an alternative schedule $\mathcal{A}'$ where no job increases its cost and at least one job reduces its cost if and only if there exists a job k which is assigned to a machine with at least one other job in $\mathcal{A}$, and moving it to an empty machine of maximum speed does not increase its cost.*

PROOF. We first assume that such a job k exists. Consider the schedule $\tilde{\mathcal{A}}$ in which k is assigned to a machine of maximum speed which is empty in $\mathcal{A}$, and the rest of the assignment is the same as in $\mathcal{A}$. There is at least one job which is assigned to the same machine as k in $\mathcal{A}$, whose cost is strictly reduced (since the load of its machine decreases when k is moved to another machine). The cost of k does not increase, and any job assigned to any machine other than the machine of k in $\mathcal{A}$ and the machine of k in $\tilde{\mathcal{A}}$ keeps its previous cost.

Next, assume that $\mathcal{A}'$ exists, and assume that among such schedules, $\mathcal{A}'$ has a minimum number of jobs which are assigned not to the same machine as in $\mathcal{A}$. Using Theorem 6, we get that $\mathcal{A}$ necessarily has an empty machine $M_{i'}$ which is non-empty in $\mathcal{A}'$. Let k be a job assigned to $M_{i'}$ in $\mathcal{A}'$ and let M_i be the machine to which it is assigned in $\mathcal{A}$.

If machine M_i does not have an additional job in $\mathcal{A}$, and since its cost on $M_{i'}$ (possibly with additional jobs) is no larger, we get $s_{i'} \geq s_i$. However, the schedule is a NE, so k cannot reduce its cost by moving to an empty machine. Therefore, its cost on $M_{i'}$ is the same as its cost on M_i, $s_{i'} = s_i$ and k is assigned to $M_{i'}$ alone in $\mathcal{A}'$. The jobs assigned to M_i in $\mathcal{A}'$ are not assigned to $M_{i'}$ or to M_i in $\mathcal{A}$. This is true since $M_{i'}$ is empty in $\mathcal{A}$ and M_i only has the job k in $\mathcal{A}$. We construct a schedule $\hat{\mathcal{A}}$ where the jobs assigned to M_i and $M_{i'}$ in $\mathcal{A}'$ are swapped and the other jobs are assigned to the same machines as in $\mathcal{A}$. The number of jobs assigned to a different machine from their machines in $\mathcal{A}$ is reduced by 1 (due to k being assigned to the same machines in $\hat{\mathcal{A}}$ and $\mathcal{A}$), which contradicts the choice of $\mathcal{A}'$.

Thus, there exists an additional job k' assigned to M_i in $\mathcal{A}$. Since moving k to some empty machine does not increase its cost, then moving it to an empty machine with maximum speed clearly does not increase its cost. □

Given the claim, if every non-empty machine has a single job then the schedule is a SPO-NE. Otherwise, the algorithm tests the existence of a job k as in the claim.

THEOREM 19. *i. The problem of checking whether a given schedule on unrelated machines is a* WPO-NE *is strongly co-NP-complete. ii. The problem of checking whether a given schedule on unrelated machines is a* SPO-NE *is co-NP-complete.*

PROOF. Given a schedule and an alternative schedule, checking whether the alternative schedule implies that the given schedule is not a NE or not (weakly or strictly) Pareto optimal can be done in polynomial time, and therefore the problems are in co-NP.

To prove hardness of the recognition of WPO-NE, we reduce from the 3-PARTITION problem, which is strongly NP-hard. In this problem we are given an integer B and $3M$ integers $a_1, a_2, \ldots, a_{3M}$, where $\frac{B}{4} < a_k < \frac{B}{2}$ for $1 \leq k \leq 3M$, $\sum_{k=1}^{3M} a_k = MB$, and we are asked whether there exists a partition of the integers into M sets, where the sum of each subset is exactly B. We construct an input with $m = 4M$ machines. There are $4M$ jobs, $3M$ of them are based on the instance of 3-PARTITION and the last M jobs are dummy jobs. For $1 \leq k \leq 3M$, we have $p_{ik} = B + 1$ for $1 \leq i \leq 3M$, and $p_{ik} = a_k$ for $3M + 1 \leq i \leq 4M$. For $3M + 1 \leq k \leq 4M$, we have $p_{ik} = B$ for $1 \leq i \leq 3M$, and $p_{ik} = B + 1$ for $3M + 1 \leq i \leq 4M$. The given schedule is one where job k is assigned to machine k. All machines have a load of $B+1$, so the schedule is a NE. We show that the schedule is weakly Pareto optimal if and only if a 3-partition as required does not exist. Assume first that a 3-partition exists. We define an alternative schedule. In this schedule, each one of the last M machines runs one subset of jobs of the first $3M$ jobs, out of the M subsets of the 3-partition. The sum of the corresponding subsets of numbers in the input of 3-PARTITION is B and therefore, their total processing time on such a machine is B. Each dummy job runs on a different machine out of the first $3M$ machines, having a cost of B. Thus, all jobs have a smaller cost in the alternative schedule, so the original one is not Pareto optimal.

On the other hand, if there exists an alternative schedule where all jobs reduce their costs, then all the first $3M$ jobs must be assigned to the last M machines (since on the other machines even if such a job is assigned to alone it still has a cost of B). For job k, no matter which such machine receives it, it has a processing time of a_k on it, so all jobs have a total processing time of MB. Since all numbers are integers, the only way that every job reduces its load is that each machine will have a load of exactly B, which implies a 3-partition.

To prove hardness of the recognition of SPO-NE, we can use the reduction of [12] showing that the recognition of SNE is hard. For completeness we present an alternative reduction. To prove hardness of the recognition of SPO-NE, we reduce from the PARTITION problem, which is NP-hard. In this problem we are given an integer B and N integers $a_1, a_2, \ldots, a_N$, where, $\sum_{k=1}^{N} a_k = 2B$, and we are asked whether there exists a partition of the integers into two sets, where the sum of each subset is exactly B. We construct an input with $m = 2$ machines (it is possible to use the same input for any larger number of machines, giving all jobs infinite processing times on every machine except for the first two machines). We have $N + 2$ jobs. Job k, for $1 \leq k \leq N$, $p_{1k} = a_k + \frac{1}{2N}$ while $p_{2k} = a_k$. Job $N + 1$ has $p_{1(N+1)} = B$ and $p_{2(N+1)} = B + \frac{1}{2}$. Job $N + 2$ has $p_{1(N+2)} = \infty$ and $p_{2(N+1)} = B$ so it must be assigned to M_2. We are given the schedule where the first N jobs are assigned to M_1 and the two last jobs are assigned to M_2. The loads of both machines are $2B + \frac{1}{2}$, thus this schedule is a NE. If there exists a partition, consider the alternative schedule where each machine receives one subset of jobs whose total size in the original input is B, and job $N + 1$ is assigned to M_1. Let K_1 be the cardinality of the set of jobs assigned to M_1 in the alternative schedule. Then the resulting load of M_1 is $2B + \frac{K_1 - 1}{2N}$. Since M_2 receives at least two jobs, then $K_1 \leq N$, so the load is strictly below $2B + \frac{1}{2}$. The load of M_2 is exactly $2B$. Thus, the original schedule is not strictly (or weakly) Pareto optimal. On the other hand, if the original schedule is not strictly (or weakly) Pareto optimal, then in an alternative schedule, job $N + 1$ must be assigned to M_1, and the total processing time of jobs assigned with it must be strictly below $B + 1$. The total processing time of jobs assigned to M_2 must be strictly below $B + 1$ as well, and so there are two sets whose sizes (in the original input) are at most B, which implies a partition.

Note that this reduction can be used to prove the (weak) co-NP-completeness of the recognition of WPO-NE schedules. Thus both problems are hard for any number of machines. □

4. CONCLUSIONS

In this paper we have studied the quality and complexity of the strict and weak Pareto optimal Nash equilibria in job scheduling games, in the settings of identical, related and unrelated machines.

We found that in the models of identical and related machines, strict and weak Pareto optimal Nash equilibria can be as bad as pure Nash equilibria, however in the model of unrelated machines, while for weak Pareto optimal Nash equilibria and $m \geq 3$ this is still the case, strict Pareto optimal Nash equilibria (and even weak Pareto optimal equilibria, for $m = 2$) are as good as Strong Nash equilibria w.r.t. the Price of Anarchy. This implies that for unrelated machines, cooperation between all players (as opposed to cooperation between subsets of players) still gives solutions of high quality.

As for identical and related machines, recognition of weakly or strictly Pareto optimal equilibria can be done in polynomial time, unlike strong equilibria. Despite the slightly worse quality of such equilibria compared to strong equilibria (due to the results for the Price of Anarchy on related machines), we conclude that weak and strict Pareto optimal equilibria are of interest for identical and related machines.

5. ACKNOWLEDGMENT

The authors would like to thank Asaf Levin for many helpful discussions.

6. REFERENCES

[1] N. Andelman, M. Feldman, and Y. Mansour. Strong price of anarchy. *Games and Economic Behavior*, 65(2):289–317, 2009.

[2] E. Anshelevich, A. Dasgupta, J. M. Kleinberg, É.Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. *SIAM Journal on Computing*, 38(4):1602–1623, 2008.

[3] R. J. Aumann. Acceptable points in general cooperative n-person games. In A. W. Tucker and R. D. Luce, editors, *Contributions to the Theory of Games IV, Annals of Mathematics Study 40*, pages 287–324. Princeton University Press, 1959.

[4] Y. Aumann and Y. Dombb. Pareto efficiency and approximate pareto efficiency in routing and load balancing games. In *Proc. of the 3rd International Symposium on Algorithmic Game Theory (SAGT'10)*, 2010.

[5] B. Awerbuch, Y. Azar, Y. Richter, and D. Tsur. Tradeoffs in worst-case equilibria. *Thoeretical Computer Science*, 361(2-3):200–209, 2006.

[6] S. Chien and A. Sinclair. Strong and pareto price of anarchy in congestion games. In *Proc. of the 36th International Colloquium on Automata, Languages and Programming (ICALP'09)*, pages 279–291, 2009.

[7] A. F. T. Committee. ATM forum traffic management specification version 4.0, 1996.

[8] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. *ACM Transactions on Algorithms*, 3(1), 2007.

[9] P. Dubey. Inefficiency of Nash equilibria. *Mathematics of Operations Research*, 11(1):1–8, 1986.

[10] L. Epstein, M. Feldman, and T. Tamir. Approximate strong equilibria in job scheduling games: an analysis for two uniformly related machines. Manuscript, 2009.

[11] E. Even-Dar, A. Kesselman, and Y. Mansour. Convergence time to Nash equilibrium in load balancing. *ACM Transactions on Algorithms*, 3(3):32, 2007.

[12] M. Feldman and T. Tamir. Approximate strong equilibrium in job scheduling games. *Journal of Artificial Intelligence Research*, 36:387–414, 2009.

[13] A. Fiat, H. Kaplan, M. Levy, and S. Olonetsky. Strong price of anarchy for machine load balancing. In *Proc. of the 34th International Colloquium on Automata, Languages and Programming (ICALP'07)*, pages 583–594, 2007.

[14] G. Finn and E. Horowitz. A linear time approximation algorithm for multiprocessor scheduling. *BIT Numerical Mathematics*, 19(3):312–320, 1979.

[15] D. Fotakis, S. C. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. G. Spirakis. The structure and complexity of Nash equilibria for a selfish routing game. *Theoretical Computer Science*, 410(36):3305–3326, 2009.

[16] T. Harks, M. Klimm, and R. H. Möhring. Strong Nash equilibria in games with the lexicographical improvement property. In *Proc. of the 5th International Workshop on Internet and Network Economics (WINE'09)*, pages 463–470, 2009.

[17] R. Holzman and N. Law-Yone. Strong equilibrium in congestion games. *Games and Economic Behavior*, 21(1-2):85–101, 1997.

[18] R. M. Karp, E. Koutsoupias, C. H. Papadimitriou, and S. Shenker. Optimization problems in congestion control. In *Proc. of 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS'00)*, pages 66–74, 2000.

[19] E. Koutsoupias, M. Mavronicolas, and P. G. Spirakis. Approximate equilibria and ball fusion. *Theory of Computing Systems*, 36(6):683–693, 2003.

[20] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS'99)*, pages 404–413, 1999.

[21] D. T. Luc. Pareto optimality. In A. Chinchuluun, P. M. Pardalos, A. Migdalas, and L. Pitsoulis, editors, *Pareto optimality, game theory and equilibria*, pages 481–515. Springer, 2008.

[22] M. Mavronicolas and P. G. Spirakis. The price of selfish routing. *Algorithmica*, 48(1):91–126, 2007.

[23] I. Milchtaich. Network topology and the efficiency of equilibrium. *Games and Economic Behavior*, 57(2):321–346, 2006.

[24] R. B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1991.

[25] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951.

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

[27] A. Skopalik and M. Hoefer. On the complexity of pareto-optimal Nash and strong equilibria. In *Proc. of the 3rd International Symposium on Algorithmic Game Theory (SAGT'10)*, pages 312–322, 2010.

[28] W. E. Walsh and M. P. Wellman. A market protocol for decentralized task allocation. In *Proc. of the 3rd International Conference on Multiagent Systems (ICMAS1998)*, pages 325–332, 1998.