Equilibria for two parallel links: The strong price of anarchy versus the price of anarchy

Leah Epstein*

Abstract

We consider a job scheduling game with two uniformly related parallel machines (or links). Jobs are atomic players, and the delay of a job is the completion time of the machine running it. The private goal of each job is to minimize its own delay and the social goal is to minimize the maximum delay of any job, that is, to minimize the makespan. We consider the well known *price of anarchy* (POA) as well as the *strong price of anarchy* (SPOA), and show that for a wide range of speed ratios these two measures are very different whereas for other speed ratios these two measures give the exact same bound. We extend all our results for models of restricted assignment, where a machine may have an initial load resulting from jobs that can only be assigned to this machine, and show tight results for all variants.

1 Introduction

A Nash equilibrium (NE) is a kind of solution concept of a game, involving two or more players, where no player can gain anything by changing only his own strategy unilaterally. If each player has chosen a strategy and no player can benefit by changing his strategy while the other players keep theirs unchanged, then the current set of strategy choices and the corresponding payoffs constitute a Nash equilibrium. If a player chooses to take one action with probability 1 then that player is playing a pure strategy, and otherwise a mixed strategy. If all players have pure strategies, the resulting equilibrium is called *pure* (see [26]).

In recent years, computer scientists started to adopt some game theoretical concepts and terminology in their studies. A large number of studies of Nash Equilibria, for problems coming from the field of computer science, were carried out in the last few years. Koutsoupias and Papadimitriou [23, 22] proposed to investigate the behavior of the worst case *coordination ratio*, which is the ratio between the social cost of the worst NE and the social optimum.

Aumann [2] was the first one to introduce a number of concepts in game theory. One of these concepts was a *strong equilibrium* (SNE), which is a pure NE, in which not only single players cannot benefit from changing their strategy (to a different pure strategy), but no non-empty subset of players can form a coalition, where a coalition means that all of them can change their strategies together, and all gain from the change (see [2, 1, 7]).

In this paper, we study pure Nash equilibria and strong equilibria for a scheduling problem on uniformly related machines. We next define the problem and the meaning of equilibria in this context.

Scheduling on uniformly related machines is a basic assignment problem. In such problems, a set of jobs $J = \{j_1, j_2, \dots, j_n\}$ is to be assigned to a set of m machines $M = \{M_1, \dots, M_m\}$, where machine M_i has a speed s_i. The size of job j_k is denoted by p_k and it is equal to its running time on a unit speed

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

machine. Moreover, the running time of this job on a machine of speed s is $\frac{p_k}{s}$. An assignment or schedule is a function $S : J \to M$. The completion time of machine M_i, which is also called the delay of this machine, is $\sum\limits_{k:S(j_k)=M_i} \frac{p_k}{s_i}$. The cost, or the *social cost* of a schedule is the maximum delay of any machine, i.e., the makespan.

In this paper we consider the case of two uniformly related machines. We assume (without loss of generality) that M_1 has unit speed and M_2 has speed $s \geq 1$. If $s = 1$, then the machines have identical speed but we still use the same notations, that is, the roles of M_1 and M_2 are fixed. We consider pure Nash equilibria and strong equilibria. The delay of a job is defined to be the delay of the machine that runs it. Seeing this scheduling problem as a game, the players are the jobs who are selfishly interested in minimizing their own delays.

A schedule is a *Nash equilibrium* (NE) if there exists no job that can decrease its delay by migrating to a different machine. More precisely, consider an assignment $S : J \to \{M_1, M_2\}$. The class of schedules $\mathcal{S}$ contains all schedules S_k that differ from S only in the assignment of j_k, that is $S_k(j_\ell) = S(j_\ell)$ for all $\ell \neq k$ and $S_k(j_k) \neq S(j_k)$, that is if $S(j_k) = M_1$ then $S_k(j_k) = M_2$ and otherwise $S_k(j_k) = M_1$. For cases where the number of machines is larger than 2, $\mathcal{S}$ contains a wider class of schedules, allowing each job to move to any machine. We say that S is a (pure) NE if for any job j_k, the delay of j_k in S_k is no smaller than its delay in S. Pure Nash equilibria do not necessary exist for all games (as opposed to mixed Nash equilibria). It is known that for scheduling games of this type, a pure NE always exists [16, 11].

A schedule is a *strong equilibrium* (SNE) if there exists no (non-empty) subset of jobs, such that if all jobs in this set migrate to a different machine simultaneously, this results in a smaller delay for each and every one of them. More precisely, given a schedule S, we can define a class of schedules $\mathcal{S}$ which contains all schedules S_K, where $K \subseteq J$, $K \neq \emptyset$. For $\ell \notin K$, we have $S_K(j_\ell) = S(j_\ell)$ whereas for $\ell \in K$, we have $S_K(j_\ell) \neq S(j_\ell)$. S is a SNE if for any $K \neq \emptyset$, there exists at least one job $j_k \in K$ whose delay in S_K is no smaller than its delay in S. A SNE is always a pure NE (by definition). Strong equilibria do not necessarily exist. Andelman, Feldman and Mansour [1] were the first to study strong equilibria in the context of scheduling and proved that scheduling games (of a more general form) admit strong equilibria. More general studies of the classes of congestion games which admit strong equilibria were studied in [20, 30].

In general, there is a recent interest in studies that separate the effect of the lack of coordination between players from the effect of their selfishness (see e.g. [17]). A NE that is not a social optimum is a stable situation not only since users are selfish, but also since the type of moves they consider is unilateral moves. Strong equilibria are stable situations whose stability is only the result of selfishness, since coordination between players is possible.

We consider the following four variants of scheduling on two uniformly related machines. The first variant is the standard one where any job can run on any machine. Three other variants relate to the so called *restricted assignment* problem. In this problem, each job is associated not only with a size, but also with a list of machines it can be processed on. This means that each job can run on one of the three subsets $\{1, 2\}$,$\{1\}$ and $\{2\}$. Thus for the case of two machines, a job can either run on any machine, or is restricted to one of the machines. Therefore, this model is equivalent to the case where machines may have an initial load that cannot switch machines. This generalization was mentioned already in the seminal paper of Koutsoupias and Papadimitriou [23]. The two additional models are the hierarchical models (see [4]), in which every job is associated with a prefix (or suffix) of the machines. In the first hierarchical model, each job is associated with one of the sets $\{1, 2\}$,$\{1\}$, whereas in the second hierarchical model the sets are

$\{1,2\}$,$\{2\}$ (if $s = 1$, only one hierarchical model exists). We therefore consider four different variants. **0.** No machine may have an initial load. **1.** Only M_1 may have initial load **2.** Only M_2 may have initial load. **3.** Any machine may have an initial load.

Let the initial load of machine i be e_i, and the total size of jobs assigned to machine i be h_i. The delay of a machine is defined to be the total size of jobs and initial loads on this machine, divided by its speed. Therefore, the delay of M_1 is $e_1 + h_1$ and the delay of M_2 is $\frac{e_2+h_2}{s}$.

In our scheduling model, the *coordination ratio*, or *price of anarchy (*POA*)* (see [28]) is the worst case ratio between the cost of a pure NE and the cost (i.e., makespan) of an optimal schedule, denoted by OPT. The *strong price of anarchy (*SPOA*)* is defined similarly, but only strong equilibria are considered. Therefore we refer to the pure price of anarchy by POA and when we discuss the mixed price of anarchy we call it the mixed POA. Note that a pure equilibrium is a special case of mixed equilibria.

We study the POA and the SPOA for all these models as functions of s. We denote the POA and SPOA for the i-th variant by $\mathrm{POA}_i(s)$ and $\mathrm{SPOA}_i(s)$.

It is noted in a series of papers (e.g., [23, 25, 27, 6, 5]) that the model which we study is a simplification of problems arising in real networks, that seems appropriate for describing basic problems in networks.

Previous work. We mention several related results for similar models of scheduling. We survey the known results for the POA and SPOA and see that in some models these measures give the same results, whereas in other models the SPOA allows to obtain more meaningful results.

The most general case is unrelated machines, where the time to run a job j_k on a machine M_i is a function of k and i. In this model the POA is unbounded [3], which holds already for a setting of two machines. Surprisingly the SPOA for this problem is bounded by the number of machines m, as shown by Fiat et al. [14], and this is tight [1]. The upper bound of 2 for two machines was already shown in [1] (an upper bound of $2m - 1$ for $m \geq 3$ was shown in that paper as well). It can be seen that in this case, separating the effect of lack of coordination from the effect of selfishness reveals a linear ratio (in the number of machines) between the cost of worst case equilibrium and the optimal cost.

Awerbuch et al. [3] focused on scheduling with restricted assignment and identical speed machines. Each job can run on only a subset of the machines, and has a fixed running time on all machines that can run it. They show that the POA is $\Theta(\frac{\log m}{\log\log m})$ (and $\Theta(\frac{\log m}{\log\log\log m})$ for mixed strategies). Their result holds for the hierarchical machines model as well, which for m machines means that the subset of allowed machines is a prefix of the machines for every job. The result for the (pure) POA appears also in [18]. Levy [24] observed that the results on the (pure) POA in this case are valid for the SPOA as well.

For m identical machines, the POA is $\frac{2m}{m+1}$ which can be deduced from the results of [15] (the upper bound) and [29] (the lower bound). It was shown in [1] that the SPOA has the same value as the POA for every m. Note, however, that the mixed POA is non constant already in this case, and equals $\Theta(\frac{\log m}{\log\log m})$, where the lower bound was shown by Koutsoupias and Papadimitriou [23] and the upper bound by Chumaj and Vöcking [6] and independently by Koutsoupias, Mavronicolas and Spirakis [21]. Tight bounds of $\frac{3}{2}$ on the mixed POA for two identical machines were shown by [23].

We conclude the survey of previous work by the known results for scheduling on uniformly related machines, the model which we study in this paper. A number of papers studied this model [23, 25, 6, 13, 14]. It is typically assumed that there is no initial load on the machines. Chumaj and Vöcking [6] showed that the POA is $\Theta(\frac{\log m}{\log\log m})$ (and $\Theta(\frac{\log m}{\log\log\log m})$ for mixed strategies). Feldmann et al. [13] proved that the POA for $m = 2$ and $m = 3$ is $\frac{\sqrt{4m-3}+1}{2}$ which equals $\phi = \frac{\sqrt{5}+1}{2}$ for two machines and 2 for three machines. They did not investigate the POA as a function of the machine speeds. As for the mixed POA, it was shown in [23] that it is at least $1 + \frac{s}{s+1}$ for $s \leq \phi$. Recently, Fiat et al. [14] showed that the SPOA for this model is $\Theta(\frac{\log m}{(\log\log m)^2})$.

2 Statement of Results

In this paper, we deal with the question of whether the difference between the POA and SPOA for uniformly related machines is a property caused by having a relatively large number of machines, or whether this is an inherent property which is true for every combination of speeds. We focus on the case of two machines with speed ratio s. We demonstrate the differences in POA and SPOA, that exist in two of the models, for a range of values of s. However, we find that in many cases the POA and SPOA are defined by the same function. Thus we show that the difference between POA and SPOA exist already for a small number of machines. However, the two measures are not different for *any* set of speeds. This result proves that the two measures are strongly related, but not identical already for relatively simple cases.

To state the results precisely, i.e., in order to specify the tight bounds on all eight functions $\mathrm{POA}_i(s)$ and $\mathrm{SPOA}_i(s)$ (for $i = 1, 2, 3, 4$), we define the following five functions.

$$F_A(s) = \begin{cases} 1 + \frac{s}{s+2}, & 1 \le s \le \sqrt{2} \approx 1.4142 \\ s, & \sqrt{2} \le s \le \phi = \frac{1+\sqrt{5}}{2} \approx 1.618 \\ 1 + \frac{1}{s}, & s \ge \phi\ , \end{cases}$$

$$F_B(s) = \begin{cases} 1 + \frac{1}{s+1}, & 1 \le s \le \sqrt{2} \\ s, & \sqrt{2} \le s \le \phi \\ 1 + \frac{1}{s}, & s \ge \phi\ , \end{cases}$$

$$F_C(s) = \begin{cases} 1 + \frac{s}{s+1}, & 1 \le s \le \phi \\ 1 + \frac{1}{s}, & s \ge \phi\ , \end{cases}$$

Let s_1 be the root of $s^3-2s^2-s+1 = 0$ in the interval $(2, 3)$, and let s_2 be the root of $3s^3-4s^2-3s+2 = 0$ in the interval $(\frac{5}{3}, 2)$.

$$G_A(s) = \begin{cases} 1 + \frac{s}{s+2}, & 1 \le s \le \sqrt{2} \\ s, & \sqrt{2} \le s \le \phi \\ \frac{1}{s-1}, & \phi \le s \le \sqrt{3} \approx 1.732 \\ 1 + \frac{1}{s+1}, & \sqrt{3} \le s \le 2 \\ \frac{s^2}{2s-1}, & 2 \le s \le s_1 \approx 2.24698 \\ 1 + \frac{1}{s}, & s \ge s_1\ , \end{cases}$$

$$G_B(s) = \begin{cases} 1 + \frac{s}{s+1}, & 1 \le s \le \phi \\ \frac{1}{s-1}, & \phi \le s \le s_2 \approx 1.69152 \\ 1 + \frac{s^2}{2s^2+s-1}, & s_2 \le s \le s_1 \\ 1 + \frac{1}{s}, & s \ge s_1\ , \end{cases}$$

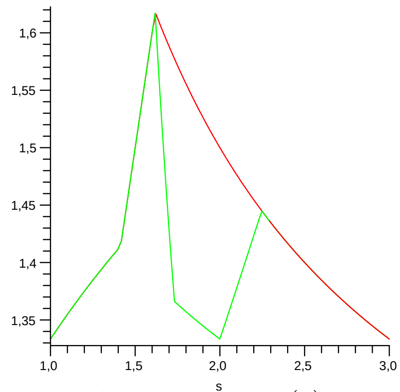

Figure 1: Results for the case without initial load: $\text{POA}_0(s) = F_A(s)$ (top) and $\text{SPOA}_0(s) = G_A(s)$ (bottom).

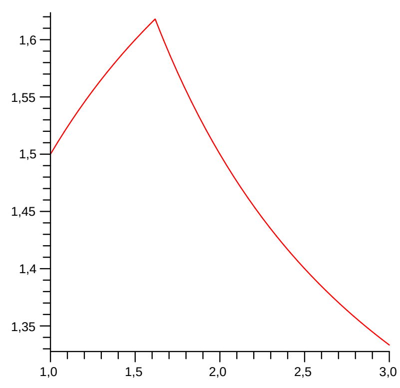

Figure 2: Results for the case with possible initial load on each machine: $\text{POA}_3(s) = \text{SPOA}_3(s) = F_C(s)$.

We prove the following theorems. A summary of the results can be found in Table 1. Graphs of the POA and SPOA functions can be found in Figures 1,2,4 and 3.

Theorem 1 *The price of anarchy as a function of $s \geq 1$ is exactly $F_A(s)$ if no initial load may exist on any of the machines, $F_B(s)$ if only M_2 may have an initial load, and $F_C(s)$ if M_1 may have an initial load.*

Theorem 2 *The Strong price of anarchy as a function of $s \geq 1$ is exactly $G_A(s)$ if no initial load may exist on any of the machines, $G_B(s)$ if M_1 may have an initial load but M_2 cannot have an initial load, $F_B(s)$ if M_2 may have an initial load but M_1 cannot have an initial load, and $F_C(s)$ if both M_2,M_1 may have an initial load.*

Note that for in the two cases where the result for the POA is different from the result for the SPOA, we get that the SPOA and POA are equal for $s \leq \phi$ and for $s > s_1$. Thus for values of s that are close to 2, the two measures are different. On the other hand, for relatively small values of s and large values of s, the two

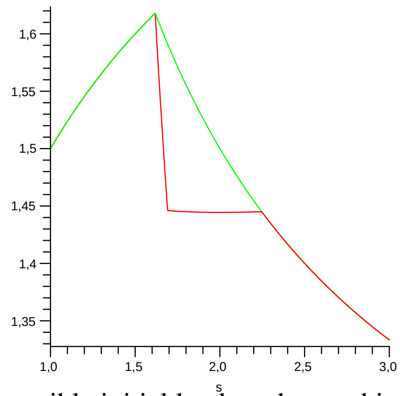

Figure 3: Results for the case with possible initial load on the machine of speed 1, M_1: $\mathrm{POA}_1(s) = F_C(s)$ (top) and $\mathrm{SPOA}_1(s) = G_B(s)$ (bottom).

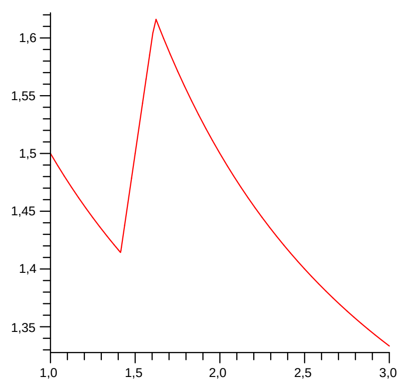

Figure 4: Results for the case with possible initial load on the machine of speed s, M_2: $\mathrm{POA}_2(s) = \mathrm{SPOA}_2(s) = F_B(s)$.

i	0	1	2	3
POA(i)	$F_A(s)$	$F_C(s)$	$F_B(s)$	$F_C(s)$
SPOA(i)	$G_A(s)$	$G_B(s)$	$F_B(s)$	$F_C(s)$

Table 1: Overview of Results

measures give the same result. In the other two cases we learn that the two measures give the same result. The overall bound (maximum value over all speeds) is ϕ in all the cases, which is achieved for $s = \phi$.

The difficulty in proving the results lies in correct identification of the different intervals for which the behavior of the POA is different. Moreover, instances which result in lower bounds on both the SPOA and POA should be distinguished from those that are valid only for the POA. Upper bounds require a careful examination of all possibilities.

We note some properties of the functions that follow from the definitions of POA and SPOA, and from the relation between the models.

Proposition 1 *1. For every i,* $\mathrm{POA}_i(s) \geq \mathrm{SPOA}_i(s)$.

2. $\mathrm{POA}_0(s) \leq \mathrm{POA}_2(s) \leq \mathrm{POA}_3(s)$, $\mathrm{POA}_0(s) \leq \mathrm{POA}_1(s) \leq \mathrm{POA}_3(s)$.

3. $\mathrm{SPOA}_0(s) \leq \mathrm{SPOA}_2(s) \leq \mathrm{SPOA}_3(s)$, $\mathrm{POA}_0(s) \leq \mathrm{SPOA}_1(s) \leq \mathrm{SPOA}_3(s)$.

For a schedule S, we use S to denote the makespan of S as well (thus OPT denotes the optimal makespan as well as an optimal assignment).

3 Lower bounds

We present the instances which allow to prove the lower bounds in Table 2. The ranges in which each instance is used can be found in Table 3.

It is not difficult to verify the next claim.

Proposition 2 *Consider each instance in the ranges for which this instance is used. All job sizes are positive, and all initial loads are non-negative. In all cases except for instance J_6, the jobs are given sorted in non-increasing order.*

Proposition 3 *The optimal makespan of each instance is as it is described in the table.*

Proof In all cases except for J_5, in the given schedule, both machines have the same delay. The instance J_5 is used for $s \in (2, s_1)$. The delay of M_2 is $2s-1$. The delay of M_1 is $s(s-1)^2 \leq 2s-1$, since this is equivalent to $s^3 - 2s^2 - s + 1 \leq 0$, which holds for $2 < s < s_1$. □

We use the following property.

Proposition 4 *Any schedule on two machines where both machines have the same delay is a pure* NE. *A schedule where machine M_i has a larger delay than the other machine is a pure* NE *if and only if the job of minimum size assigned to M_i cannot reduce its cost by moving to the other machine.*

Using this proposition, we prove the following for the schedules defined for the instances of Table 2.

Lemma 1 *Each one of the schedules S_i of Table 2 is a pure* NE.

Proof It is not hard to verify the delays of the machines in the schedules S_i ($1 \leq i \leq 10$), and that the machine of larger delay is the one whose delay is written in bold. For all schedules except for S_2 and S_3, the resulting delay of the machine of smaller delay, if a job of minimum size from the other machine joins it, is the same as the delay of the machine with larger delay, so the job would not decrease its cost. For S_2, which is used for $\sqrt{2} \leq s \leq \phi$, this resulting delay is $1 + \frac{1}{s} \geq s$ (which holds for $s \leq \phi$). For S_3, which is used for $\phi \leq s \leq \sqrt{3}$, this resulting delay is $s - \frac{1}{s} \geq 1$, (which holds for $s \geq \phi$). □

Proposition 5 *Given a schedule on two machines which is a pure* NE*, if this schedule is not a* SNE*, then a coalition of jobs where every job can reduce its cost consists of at least one job of each one of the machines. Moreover, the coalition cannot contain the entire set of jobs coming from M_2.*

	Job sizes	Initial load	Optimal makespan	OPT	delays in S_i	S_i
J_1	$s^2+s, s^2+s, s, 2-s^2$	0	$s+2$	$\{1,4\},\{2,3\}$	$2+s-s^2$, $\mathbf{2s+2}$	$\{3,4\},\{1,2\}$
J_2	$s, 1$	0	1	$\{2\},\{1\}$	$\mathbf{s}$, $\frac{1}{s}$	$\{1\},\{2\}$
J_3	$1, s-1, s^2-s-1$	0	$s-1$	$\{2\},\{1,3\}$	$\mathbf{1}$, $s-\frac{2}{s}$	$\{1\},\{2,3\}$
J_4	$s^2+s-1, s+1, 1$	0	$s+1$	$\{2\},\{1,3\}$	1, $\mathbf{s+2}$	$\{3\},\{1,2\}$
J_5	$s^2, s(s-1)^2, s^2-s$	0	$2s-1$	$\{2\},\{1,3\}$	$\mathbf{s^2}$, s^2-s	$\{1\},\{2,3\}$
J_6	$s+1, s, s^2-s-1$	0	s	$\{2\},\{1,3\}$	$\mathbf{s+1}$, $s-\frac{1}{s}$	$\{1\},\{2,3\}$
J_7	s^2+s, s^2	$s+1-s^2$ (1)	$s+1$	$\{2\},\{1\}$	$\mathbf{2s+1}$, s	$\{1\},\{2\}$
J_8	s^3+s^2, s^3, s^3-s	$2s^2+s-1-s^3$ (1)	$2s^2+s-1$	$\{2\},\{1,3\}$	$\mathbf{3s^2+s-1}$, $2s^2-1$	$\{1\},\{2,3\}$
J_9	$s+1, 1$	s^2+s-1 (2)	$s+1$	$\{1\},\{2\}$	1, $\mathbf{s+2}$	$\{2\},\{1\}$
J_{10}	$s+1, s$	s^2-s-1 (2)	s	$\{2\},\{1\}$	$\mathbf{s+1}$, $s-\frac{1}{s}$	$\{1\},\{2\}$

Table 2: The instances which are used to prove the lower bounds. The first column contains the names of the instances. The second column contains the jobs while the third column contains the initial loads, if exist, where the number in parenthesis is the index of the machine which has the initial load. The next two columns contain the optimal makespan and the parition into two sets (assigned to M_1 and M_2 respectively) which allows it. The last two columns contain for each instance J_i the delay of each machine in an assignment S_i (the delay which appears in bold is the larger delay) and the partition which defines it.

	SPOA$_0$	POA$_0$	SPOA$_1$	POA$_1$	SPOA$_2$	POA$_2$	SPOA$_3$	POA$_3$
$[1, \sqrt{2} \approx 1.414)$	J_1	J_1	J_7	J_7	J_9	J_9	J_7	J_7
$[\sqrt{2}, \phi \approx 1.618]$	J_2	J_2	J_7	J_7	J_2	J_2	J_7	J_7
$(\phi, s_2 \approx 1.691]$	J_3	J_6	J_3	J_6	J_{10}	J_6, J_{10}	J_{10}	J_6, J_{10}
$(s_2, \sqrt{3} \approx 1.732]$	J_3	J_6	J_8	J_6	J_{10}	J_6, J_{10}	J_{10}	J_6, J_{10}
$(\sqrt{3}, 2]$	J_4	J_6	J_8	J_6	J_{10}	J_6, J_{10}	J_{10}	J_6, J_{10}
$(2, s_1 \approx 2.246)$	J_5	J_6	J_8	J_6	J_{10}	J_6, J_{10}	J_{10}	J_6, J_{10}
$[s_1, \infty)$	J_6	J_6	J_6	J_6	J_6, J_{10}	J_6, J_{10}	J_6, J_{10}	J_6, J_{10}

Table 3: The inputs which give the lower bounds for each interval of speed ratios, for each each measure.

Proof If there is a coalition which consists of jobs coming from one machine, then each one of these jobs would reduce its cost if it changes its action unilaterally, so the schedule cannot be a NE. If the entire set of jobs assigned to M_2 moves to M_1 then their delay cannot decrease, even if in the resulting schedule no additional jobs are assigned to M_1. □

Proposition 6 *Given a schedule on two machines which is a pure* NE *but not a* SNE*, then the delay of M_1 is strictly larger than the delay of M_2.*

Proof Consider such a schedule S, and consider a coalition where every job can reduce its cost. Let $Y_i + X_i$ be the total size of jobs assigned to M_i in S, where X_i is the total size of jobs in the coalition and Y_i the total size of jobs not in the coalition. By proposition 5, $X_i > 0$ for $i = 1, 2$. Since all jobs in the coalition reduce their costs, we have $\frac{X_1+Y_2}{s} < X_1 + Y_1$ and $X_2 + Y_1 < \frac{X_2+Y_2}{s}$. Multiplying the first inequality by s and summing with the second, we get $X_1 + Y_2 + X_2 + Y_1 < sX_1 + sY_1 + \frac{X_2}{s} + \frac{Y_2}{s}$. Reorganizing and dividing by $s-1$ gives $\frac{Y_2+X_2}{s} < X_1 + Y_1$. □

Lemma 2 *All the schedules of Table 2 except for S_6 are strong equilibria in all intervals where they are used. S_6 is a* SNE *for $s \in [s_1, \infty)$.*

Proof By Proposition 5, a schedule, which is a NE and M_2 in which has a single job assigned to it, is a SNE. Thus S_2, S_7, S_9 and S_{10} are strong equilibria. By Proposition 6, S_1 and S_4 are strong equilibria as well, since M_2 has a larger delay in those schedules.

For S_3, there are two possible coalitions, which are $\{j_1, j_2\}$ and $\{j_1, j_3\}$. If the first coalition deviates, the delay of j_2 would become $s - 1 > s - \frac{2}{s}$, which holds for $s < 2$, so this deviation is impossible. If the second coalition deviates, the delay of j_1 would remain 1, so this deviation is not possible either.

For S_5, in a schedule resulting of a deviation of coalition, M_1 will have one of the jobs j_2 and j_3, so its delay would be at least $s^2 - s$, so such a deviation is impossible.

For S_6, we need to consider the coalitions $\{j_1, j_2\}$ and $\{j_1, j_3\}$. Job j_2 would have a delay of s on M_1 so it does not benefit from moving. Job j_3 benefits from moving to M_1 only if $s^2 - s - 1 < s - \frac{1}{s}$, or $s^3 - 2s^2 - s + 1 < 0$, which does not hold for any $s \geq s_1$.

For S_8, the delay of M_1 if one of j_2 and j_3 would be assigned to it would be at least $2s^2 - 1$, so no coalition can exist. □

We have therefore proved the following theorem (using Proposition 1).

Theorem 3 *The instances of Table 2 imply the lower bounds as indicated in Table 1.*

4 Upper bounds

In all the proofs of this section, we assume without loss of generality that the sum of sizes of all jobs and initial loads (if exist) is exactly 1. We consider a specific optimal assignment OPT, and given the total size of jobs and initial loads, we have OPT $\geq \frac{1}{s+1}$. In each proof we consider an assignment S, which is not optimal, and intend to prove that its delay is at most $f(s) \cdot$ OPT for a function $f(s)$. To prove this, we will assume by contradiction that the maximum delay of S is larger than $f(s) \cdot$ OPT $\geq \frac{f(s)}{s+1}$. Unless stated otherwise, it is assumed that S is a NE. If it is also assumed that it is a SNE, then this assumption is stated. In what follows, d denotes the delay of S.

Lemma 3 *In S, a machine with a maximum delay has at least one job which is assigned to the other machine in* OPT. *This holds even if the machines may have initial load. If in S the machine M_2 has no initial load, and the maximum delay is achieved for M_2, then in this schedule, M_2 also has at least one job assigned to M_2 in* OPT. *This holds even if M_1 may have initial load. If in S the machine M_1 has no initial load, the maximum delay is achieved for M_1, and this delay is above $s \cdot$* OPT, *then in this schedule, M_1 also has at least one job assigned to M_1 in* OPT. *This holds even if M_2 has initial load.*

Proof In general, even if machines may have initial loads, if M_i has a maximum delay in S and it only contains (in addition to a possible initial load) the jobs which this machine contains in OPT, then S is optimal, since we get $d \leq$ OPT. Therefore, M_i contains in S a job which is assigned to the other machine by OPT.

Let d_1 and d_2 denote the delays of M_1 and M_2 (respectively) in OPT. Since S is not optimal, we have $d > d_1$ and $d > d_2$. The total size of jobs assigned to M_2 in OPT is at most $s \cdot d_2$, and for M_1 it is at most

d_1. If d is the delay of M_2, and M_2 has no initial load, then the total size of jobs assigned to M_2 in S is $s \cdot d$. Since $s \cdot d > d_1$, there must be a job coming from M_2 in OPT. If $d > s \cdot \text{OPT}$ is the delay of M_1, and M_1 has no initial load, since $s \cdot d_2 \leq s\text{OPT}$, then M_1 must have a job coming from M_1 in OPT. □

Lemma 4 *If in S, the maximum delay d is achieved for M_i, then any job assigned to M_i in this schedule has a size of at least $d(s+1)-1$. This holds even if the machines may have initial loads.*

Proof If $i = 1$, then the delay of M_2 is $\frac{1-d}{s}$. Consider a job of size y assigned to M_1 in S. If this job is moved to M_2 then its delay becomes $\frac{1-d+y}{s}$. Since S is a NE, it must satisfy $\frac{1-d+y}{s} \geq d$. If $i = 2$, then the delay of M_1 is $1 - d \cdot s$. Consider a job of size x assigned to M_2 in S. If this job is moved to M_1 then its delay becomes $1 - d \cdot s + x$. Since S is a NE, it must satisfy $1 - d \cdot s + x \geq d$. □

Lemma 5 *The delay of M_2 in every schedule (not necessarily a* NE*) is at most $\frac{1}{s}$. The delay of M_1 in S is at most $\frac{1}{s}$. In S, the machine of larger delay has at least one job assigned to it. These properties hold even if there may be initial loads on the machines.*

Proof The first part of Lemma 3 implies that the machine of larger delay has at least one job.

Since the sum of sizes of all jobs and the initial load is 1, in any possible schedule, the delay of M_2 is at most $\frac{1}{s}$. Assume next that the delay of M_1 in S is strictly above $\frac{1}{s}$ (and thus the delay of M_2 is strictly below $\frac{1}{s}$). Since M_1 has a job assigned to it, then moving such a job to M_2 still results in a delay of at most $\frac{1}{s}$ on M_2. This contradicts S being a NE. □

The following lemma follows directly from Lemma 5, since $\text{OPT} \geq \frac{1}{s+1}$.

Lemma 6 *For any $s \geq 1$, $S \leq \frac{s+1}{s}\text{OPT}$. This holds even if the machines may have an initial load.*

Lemma 7 *If in S, the maximum delay d is achieved on M_1, and $d > \frac{2}{2s+1}$, then M_1 has exactly one job assigned to it. This holds even if the machines may have initial loads. Under the same condition, if M_1 has no initial load, then the delay must satisfy $d \leq s\text{OPT}$. This holds even if M_2 may have an initial load.*

Proof By Lemma 5, M_1 has at least one job assigned to it in S. By Lemma 5, $d \leq \frac{1}{s}$. By Lemma 4, the size of each job assigned to M_1 in S is at least $d(s+1)-1$. If there are two jobs assigned to M_i, then we get $2(d(s+1)-1) \leq d$, or equivalently, $d(2s+1) \leq 2$, which does not hold for $d > \frac{2}{2s+1}$. By Lemma 3, if M_1 has no initial load and $d > s\text{OPT}$ then M_1 has at least two jobs assigned to it in S, so we conclude that $d \leq s\text{OPT}$. □

Lemma 8 *For any $s \geq 1$, if in S the machine M_2 does not contain any initial load, and the delay of M_2 is no smaller than the delay of M_1, we have $S \leq \frac{s+2}{s+1}\text{OPT}$. This holds even if M_1 may have an initial load.*

Proof Let L be the set of jobs assigned to M_2 in S and $\ell = d \cdot s$ be their total size (recall that M_2 does not have an initial load). By Lemma 3, L contains at least two jobs, one of which is assigned to M_2 in OPT and the other one is assigned to M_1 in OPT. Recall that we assume that we assume $d = \frac{\ell}{s} > \frac{(s+2)}{(s+1)^2}$. By Lemma 4, any job in L has a size of at least $(s+1)\frac{\ell}{s} - 1 > \frac{s+2}{s+1} - 1 = \frac{1}{s+1}$.

Let x be a size of a job in L which is assigned to M_1 in OPT. Since S is a NE, then $1 - \ell + x \geq \frac{\ell}{s}$. Using $\text{OPT} \geq x$ (since OPT runs this job on a machine of speed 1) and $\frac{\ell}{s} > \frac{s+2}{s+1}\text{OPT}$, we get $1 - \ell + x \geq x \cdot \frac{s+2}{s+1}$. Reorganizing we have $1 - \ell \geq \frac{x}{s+1}$. Using the bound $\ell > \frac{s(s+2)}{(s+1)^2}$ gives $\frac{1}{(s+1)^2} > \frac{x}{s+1}$, or equivalently, $x < \frac{1}{s+1}$, which is a contradiction. □

Lemma 9 *Consider a value of s such that $\phi \le s \le s_1$. If S is a* SNE*, and no machine has an initial load, then $S \le G_A(s)$*OPT.

Proof In this interval we have

$$G_A(s) = \max\left\{\frac{1}{s-1}, \frac{s+2}{s+1}, \frac{s^2}{2s-1}\right\}.$$

If M_2 has a delay which is no smaller than the delay of M_1, using Lemma 8 we have $S \le \frac{s+2}{s+1}\text{OPT} \le G_A(s)\text{OPT}$. We therefore assume that M_1 has a delay which is larger than the delay of M_2. Let L be the set of jobs assigned to M_1 and $\ell > \frac{1}{s+1}$ be their total size. We assume $\ell = d > \frac{s+2}{(s+1)}\text{OPT} \ge \frac{s+2}{(s+1)^2}$, $\ell > \frac{\text{OPT}}{s-1}$ and $\ell > \frac{s^2}{2s-1}\text{OPT}$. By Lemma 5, $\ell \le \frac{1}{s}$.

Note that $\frac{s+2}{(s+1)^2} > \frac{2}{2s+1}$, so by Lemma 7, L consists of a single job. By Lemma 3, OPT assigns this job to M_2. Let A_i be the set of jobs assigned to M_2 in S which are assigned by OPT to M_i, and let a_i be the sum of sizes of jobs in A_i. We have $\text{OPT} \ge \frac{a_2+\ell}{s}$ and $\text{OPT} \ge a_1$. Therefore, using $\ell > \frac{\text{OPT}}{s-1}$, we get $a_1 < (s-1)\ell$.

In addition, we have $\ell > \frac{s^2}{2s-1}\text{OPT} > \frac{s^2}{2s-1} \cdot \frac{a_2+\ell}{s} = (a_2+\ell)\frac{s}{2s-1}$. This gives $s \cdot a_2 < (s-1)\ell$. If we have $(s-1)a_2 \ge a_1$, then we get $1 = a_1 + a_2 + \ell \le sa_2 + \ell < s\ell \le 1$, which is a contradiction. Therefore, it is left to take care of the case $(s-1)a_2 < a_1 < (s-1)\ell$. In this case consider the coalition that consists of all jobs of $A_2 \cup L$. If each one of these jobs moves to a different machine (that is, the jobs of A_2 to M_1 and the job of L to M_2), we have that the delay of job of L changes from ℓ to $\frac{a_1+\ell}{s} < \ell$. Since S is a SNE, then the delay of jobs in A_2 cannot decrease. This delay changes from $\frac{a_1+a_2}{s}$ to a_2. However $\frac{a_1+a_2}{s} > a_2$, which is a contradiction.. □

Lemma 10 *For $1 \le s \le \phi$, if M_1 does not contain any initial load, and M_2 has a delay no smaller than M_1, then $S \le \frac{s+2}{s+1}$*OPT. *This holds even if M_2 may have an initial load.*

Proof Let b_1 denote the total size of jobs assigned to M_1 both in S and in OPT, b_2 denotes the remaining size of jobs assigned to M_1 in S, a_1 denotes the total size of jobs assigned to M_2 is S but to M_1 in OPT, and a_2 denotes the total size of jobs and the initial load assigned to M_2 in both schedules. Thus, $d = \frac{a_1+a_2}{s}$. We have $a_1 > 0$ since otherwise, using $\text{OPT} \ge \frac{a_2+b_2}{s} \ge \frac{a_2}{s} = d$ we get that S is optimal, which contradicts out assumption.

Since no job on M_2 can benefit from moving, given a job of size y assigned to M_2 in S, $b_1 + b_2 + y \ge \frac{a_1+a_2}{s}$. Let y be a minimum size of a job. Since $a_1 > 0$, $0 < y \le a_1$ must hold, and we get $sb_1 + sb_2 + (s-1)a_1 - a_2 \ge 0$. Since $a_1 + a_2 + b_1 + b_2 = 1$ this is equivalent to $s(1 - a_1 - a_2) + (s-1)a_1 - a_2 \ge 0$ or to $a_1 + (s+1)a_2 \le s \le s(s+1)\text{OPT}$ (using $\text{OPT} \ge \frac{1}{s+1}$). Since $\text{OPT} \ge b_1 + a_1 \ge a_1$ we have $sa_1 \le s\text{OPT}$. Taking the sum of the last two inequalities on a_1, a_2 we get $(s+1)(a_1+a_2) \le (s^2+2s)\text{OPT}$ which gives $\frac{d}{\text{OPT}} \le \frac{s+2}{s+1}$. □

Lemma 11 *For $1 \le s \le \sqrt{2}$, if M_1 does not contain any initial load, then $S \le \frac{s+2}{s+1}$*OPT. *This holds even if M_2 may have an initial load.*

Proof By Lemma 10, if the delay of M_2 is no smaller than the delay of M_1 we are done. Otherwise, assume that the delay of M_1 is larger than the delay of M_2.

We use the notations a_1, a_2, b_1, b_2 as in the proof of Lemma 10. We have $b_2 > 0$ since otherwise $\text{OPT} \ge a_1 + b_1 \ge b_1 = b_1 + b_2 = d$. We also have $b_1 > 0$, since otherwise we have $d = b_1 + b_2 = b_2 \le a_2 + b_2 \le s\text{OPT} \le \frac{s+2}{s+1}\text{OPT}$, since $s \le \frac{s+2}{s+1}$ for $s \le \sqrt{2}$. Thus M_1 has at least two jobs. We use Lemma 7 as follows. $b_1 + b_2 > \frac{s+2}{(s+1)^2} > \frac{2}{2s+1}$, so M_1 can only have one job, which is a contradiction. □

Lemma 12 *For $\sqrt{2} \le s \le \phi$, if M_1 does not contain any initial load, then $S \le s\text{OPT}$. This holds even if M_2 may have an initial load.*

Proof If M_2 has a delay no smaller than the delay of M_1, by Lemma 10 we get $S \le \frac{s+2}{s+1}\text{OPT} \le s\text{OPT}$. If M_1 has a delay larger than the one of M_2, let ℓ be the total size of jobs assigned to M_1, where $\ell = d$, since M_1 has no initial load. We have $\ell > s \cdot \text{OPT} \ge \frac{s}{s+1}$. We use Lemma 7 and get $\ell > \frac{s}{s+1} > \frac{2}{2s+1}$, which holds for $s > 1.29$. Thus we get $\ell \le s\text{OPT}$ which is a contradiction. $\square$

Lemma 13 *For $1 \le s \le \sqrt{2}$, if no machine contains any initial load, then $S \le (1 + \frac{s}{s+2})\text{OPT}$.*

Proof The maximum delay satisfies $d > \frac{2s+2}{s+2}\text{OPT} \ge \frac{2}{s+2}$. We start with the case where M_1 has a delay no smaller than M_2. Since for $s \le \sqrt{2}$, $\frac{2s+2}{s+2} \ge s$, we get $d > s\text{OPT}$. We use Lemma 7. We have $d > \frac{2}{s+2} \ge \frac{2}{2s+1}$, and so $d \le s\text{OPT}$ which is a contradiction.

Consider next the case where M_2 has a larger delay than the delay of M_1. Let L be the set of jobs assigned to M_2 in S. By Lemma 3, $|L| \ge 2$. By Lemma 4, the size of each job is at least $d(s+1) - 1 > \frac{2(s+1)}{s+2} - 1 = \frac{s}{s+2}$. Therefore, since $\frac{3s}{s+2} \ge 1$, there cannot be three jobs and so $|L| = 2$. Let $x \ge y > \frac{s}{s+2}$ denote the sizes of these jobs. Since S is a NE, moving the job of size x to M_1 does not reduce its delay. After moving this job, only the job of size y remains assigned to M_2. We have $1 - y \ge d > \frac{2}{s+2}$, that is, $y < \frac{s}{s+2}$, which is a contradiction. $\square$

Lemma 14 *For $s \le \phi$, $S \le \frac{2s+1}{s+1} \cdot \text{OPT}$. This holds even if the machines may have an initial load.*

Proof Let M_i be a machine with maximum delay in S. By Lemma 5, there is at least one job assigned to M_i in S. Let x be the size of a such a job j. Remove j from the schedule and let ℓ_i denote the sum of job sizes and the initial load (if exists) on M_1 and on M_2, respectively. j is assigned to a machine where its delay is minimized (otherwise it would benefit from moving). If $i = 1$, then we have $d = \ell_1 + x$ and $\frac{\ell_2+x}{s} \ge \ell_1 + x$. Otherwise we have $d = \frac{\ell_2+x}{s}$ and $\ell_1 + x \ge \frac{\ell_2+x}{s}$.

In both cases we get $\ell_1 + \ell_2 + 2x \ge (s+1)d$. On the other hand $\text{OPT} \ge \frac{\ell_1+\ell_2+x}{s+1}$ and $\text{OPT} \ge \frac{x}{s}$, thus $\ell_1 + \ell_2 + 2x \le (2s+1)\text{OPT}$. We get $\frac{d}{\text{OPT}} \le \frac{2s+1}{s+1}$. $\square$

Lemma 15 *Assume that S is a* SNE, *and M_2 has no initial load. For $s \in (\phi, s_1)$, $S \le G_B(s) \cdot \text{OPT}$. This holds even if M_1 may have an initial load.*

Proof In the interval $s \in (\phi, s_1)$ we need to show

$$\text{SPOA}(s) \le \max\left\{\frac{1}{s-1}, \frac{3s^2+s-1}{2s^2+s-1}\right\} .$$

We use the notations a_1, a_2, b_1, b_2 as in the proof of Lemma 10, where the initial load of M_1 (if exists) is included in b_1. The set of jobs whose sum is a_i (respectively b_i) is denoted by A_i (respectively B_i). We have $d > \frac{\text{OPT}}{s-1}$ and $d > \frac{3s^2+s-1}{2s^2+s-1} \cdot \text{OPT}$.

Consider first the case that d is the delay of M_1. We have $b_2 > 0$ since otherwise $d = b_1 + b_2 = b_1 \le b_1 + a_1 \le \text{OPT}$. In addition, we show $a_2 > 0$. Assume $a_2 = 0$. Since it is not beneficial for any job $j \in B_2$ of size x to move, we get $b_1 + b_2 \le \frac{a_1+x}{s} \le \frac{a_1+b_2}{s}$. Since $\text{OPT} \ge a_1 + b_1$ we get $s(b_1+b_2) \le b_2 + \text{OPT} - b_1 \le b_1 + b_2 + \text{OPT}$ which gives $b_1 + b_2 \le \frac{\text{OPT}}{s-1}$. We are therefore left with the case where $b_2 > 0$ and $a_2 > 0$, so $B_2 \ne \emptyset$ and $A_2 \ne \emptyset$.

We next consider the coalition $B_2 \cup A_2$, and the deviation where each one of these jobs switches to the other machine. Since S is a SNE, we either have that $\frac{a_1+a_2}{s} \leq b_1 + a_2$ (it is not beneficial for a job in A_2 to move) or $b_1 + b_2 \leq \frac{a_1+b_2}{s}$ (it is not beneficial for a job in B_2 to move). We already saw that the second case cannot hold, so the first option must hold. It is also not beneficial for only the jobs of B_2 to move, thus $b_1+b_2 \leq \frac{b_2+a_1+a_2}{s} = \frac{1-b_1}{s}$. We multiply the first inequality, which is equivalent to $-sb_1+a_1+(1-s)a_2 \leq 0$ or to $-b_1 + (s-1)b_2 + sa_1 \leq s-1$ (using $a_2 = 1 - b_1 - b_2 - a_1$) by 1, and the second one, which is equivalent to $(s+1)b_1 + sb_2 \leq 1$ by $2s$ and get $b_1(2s^2 + 2s - 1) + (2s^2 + s - 1)b_2 + sa_1 \leq 3s - 1$. Using $\text{OPT} \geq \frac{a_2+b_2}{s} = \frac{1-b_1-a_1}{s}$ we get $a_1 \geq 1 - s\text{OPT} - b_1$. Multiplying by $-s$ and summing up with the previous inequality we have $(2s^2 + 2s - 1)b_1 + (2s^2 + s - 1)b_2 \leq 3s - 1 - s + s^2\text{OPT} + sb_1$. Thus $S = b_1 + b_2 \leq \frac{2s-1+s^2\text{OPT}}{2s^2+s-1}$. Using $\text{OPT} \geq \frac{1}{s+1}$ we get $S \leq \frac{(2s-1)(s+1)\text{OPT}+s^2\text{OPT}}{2s^2+s-1} = \frac{3s^2+s-1}{2s^2+s-1}\text{OPT}$.

Consider the case where the delay of M_2 is no smaller than the delay of M_1. By Lemma 8, the delay of M_2 is at most $\frac{s+2}{s+1}\text{OPT}$. Since for any $s \geq 1$ we have $\frac{s+2}{s+1} \leq \frac{3s^2+s-1}{2s^2+s-1}$, we are done. □

	SPOA_0	POA_0	SPOA_1	POA_1	SPOA_2	POA_2	SPOA_3	POA_3
$[1, \sqrt{2} \approx 1.414)$	13	13	14	14	11	11	14	14
$[\sqrt{2}, \phi \approx 1.618]$	12	12	14	14	12	12	14	14
$(\phi, s_2 \approx 1.691]$	9	6	15	6	6	6	6	6
$(s_2, \sqrt{3} \approx 1.732]$	9	6	15	6	6	6	6	6
$(\sqrt{3}, 2]$	9	6	15	6	6	6	6	6
$(2, s_1 \approx 2.246)$	9	6	15	6	6	6	6	6
$[s_1, \infty)$	6	6	6	6	6	6	6	6

Table 4: The indices of lemmas required to obtain the upper bound for each case.

We have proved the following theorem. Table 4 summarizes which lemmas need to be applied to achieve each result (in addition to the usage of Proposition 1).

Theorem 4 *The upper bounds indicated in Table 1 are valid.*

5 Concluding remarks

We considered equilibria in scheduling games on two related machines and answered the question whether the price of anarchy is equal to the strong price of anarchy. Surprisingly, we find that the answer depends on the exact speed ratio, since the answer is negative for some speed ratios and positive for others. Specifically, the values of s for which the POA is larger than the SPOA in some cases are the "intermediate" values of s. A possible reason for this is that when s is small, the two machines are almost equal in speed, and if the schedule is a Nash equilibrium, then the situation where a coalition of jobs can be formed, such that each of them can reduce its delay, is more rare. Similarly, if s is large, then one of the machines is the dominant one, and the other one is much slower. Note however, that the property that every Nash equilibrium is a strong equilibrium is true only for $s = 1$ [12, 8].

We extended our results for cases where machines may have initial loads, and encountered a similar situation in one of the cases. In other two variants we found that the two measures give the same result. Thus we reconsidered the suggestion of [1] to study the SPOA rather than the POA, taking into account the possibility that players can form coalitions. The conclusion is again that in some cases the quality of equilibria improves quite significantly (for example, in the case $s = \sqrt{3} \approx 1.73$, in the case without initial load, it drops from approximately 1.577 to approximately 1.366), while in other cases, it remains relatively high (for example, in the case $s = \phi \approx 1.618$, in the case without initial load, it is approximately 1.618).

Even though the POA and SPOA as a function of the number of machines are known (up to a constant multiplicative factor) [23, 21, 6, 14], a number of other parameters are of interest. Such parameters, which often give meaningful results, can be the number of distinct speeds [10] or the ratio of largest speed to the smallest speed [9]. It was shown in [10] that the POA, as a function of the number of distinct speeds p, is exactly $p + 1$, while the lower bound on the SPOA, implied by the results of [14] is $\Omega(\frac{p}{\log p})$. The current study is involved with finding the POA and SPOA as a function of the maximum speed ratio between machines.

In this paper we only considered pure equilibria. Even though mixed strong equilibria do not necessarily exist [1] (and so strong equilibria are only defined to be pure equilibria), mixed equilibria do exist [26], and it is interesting to find the tight mixed price of anarchy as a function of the speed ratio. The current status of the problem is that even the overall bound is not known. Koutsoupias and Papadimitriou [23] conjectured that the overall upper bound is ϕ and the function for $s \leq \phi$ is $1 + \frac{s}{s+1}$, and proved a lower bound. The questions whether this is indeed the tight bound on the mixed POA in this interval, whether ϕ is the overall bound for the mixed POA, and finally, what is the exact mixed POA as a function of the speed ratio s, remain open (see the survey of Heydenreich, Müller and Uetz [19] for details).

References

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

[2] 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.

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

[4] A. Bar-Noy, A. Freund, and J. Naor. On-line load balancing in a hierarchical server topology. *SIAM Journal on Computing*, 31(2):527–549, 2001.

[5] A. Czumaj. Selfish routing on the internet. In J. Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, chapter 42. CRC Press, 2004.

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

[7] A. Epstein, M. Feldman, and Y. Mansour. Strong equilibrium in cost sharing connection games. *Games and Economic Behavior*, 67(1):51–68, 2009.

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

[9] L. Epstein, E. Kleiman, and R. van Stee. Maximizing the minimum load: The cost of selfishness. In *5th International Workshop on Internet and Network Economics (WINE2009)*, pages 232–243, 2009.

[10] L. Epstein and R. van Stee. The price of anarchy on uniformly related machines revisited. In *Proc. of the First International Symposium on Algorithmic Game Theory (SAGT2008)*, pages 46–57, 2008.

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

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

[13] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Nashification and the coordination ratio for a selfish routing game. In *Proc. of the 30th International Colloquium on Automata, Languages and Programming (ICALP2003)*, pages 514–526, 2003.

[14] 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 (ICALP2007)*, pages 583–594, 2007.

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

[16] 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.

[17] D. Fotakis, S. C. Kontogiannis, and P. G. Spirakis. Atomic congestion games among coalitions. *ACM Transactions on Algorithms*, 4(4), 2008.

[18] M. Gairing, T. Lücking, M. Mavronicolas, and B. Monien. The price of anarchy for restricted parallel links. *Parallel Processing Letters*, 16(1):117–132, 2006.

[19] B. Heydenreich, R. Müller, and M. Uetz. Games and mechanism design in machine scheduling - an introduction. To appear in *Production and Operations Management*, 2007.

[20] R. Holzman and N. Law-Yone. Strong equilibrium in congestion games. *Games and Economic Behavior*, 21(1-2):85101, 1997.

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

[22] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3:65–69, 2009.

[23] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *Proc. of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS1999)*, pages 404–413, 1999.

[24] M. Levy. Private communication, 2007.

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

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

[27] C. H. Papadimitriou. Algorithms, games, and the internet. In *Proc. of the 33rd Annual ACM Symposium on Theory of Computing (STOC2001)*, pages 749–753, 2001.

[28] T. Roughgarden. *Selfish routing and the price of anarchy*. MIT Press, 2005.

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

[30] M. Tennenholtz and O. Rozenfeld. Strong and correlated strong equilibria in monotone congestion games. In *Proc. of the 2nd International Workshop on Internet and Network Economics (WINE2006)*, page 7486, 2006.