Discrete Applied Mathematics 161 (2013) 1843–1858

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Approximate strong equilibria in job scheduling games with two uniformly related machines

Leah Epstein [a], Michal Feldman [b,c], Tami Tamir [d,*], Łukasz Witkowski [e], Marcin Witkowski [e]

[a] Department of Mathematics, University of Haifa, Israel
[b] School of Business Administration, Hebrew University of Jerusalem, Israel
[c] School of Engineering and Applied Sciences, Harvard University, United States
[d] School of Computer Science, The Interdisciplinary Center, Herzliya, Israel
[e] Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań, Poland

ARTICLE INFO

Article history:
Received 13 September 2011
Received in revised form 22 February 2013
Accepted 26 February 2013
Available online 23 March 2013

Keywords:
Scheduling
Games
Approximate strong Nash equilibrium

ABSTRACT

We consider approximate strong equilibria (SE) in strategic job scheduling games with two uniformly related machines. Jobs are assigned to machines, and each job wishes to minimize its cost, given by the completion time of the machine it is assigned to. Finding a Nash equilibrium (NE) in this game is simple. However, NE-configurations are not stable against coordinated deviations of several jobs. Various measures can be used to evaluate how well an NE-configuration approximates SE.

A schedule is said to be an α-SE if there is no coalitional deviation such that every member of the coalition reduces its cost by a factor greater than α. We show that any pure NE on two related machines of speed ratio s is a $\frac{s^2+s-1}{s^2} \leq \frac{5}{4}$-SE, and provide a matching lower bound. In addition, we show that the LPT (Longest Processing Time) algorithm provides a better approximation ratio than a general NE, in particular, any LPT schedule is a 1.1011-SE. This is in contrast to the LS (List Scheduling) greedy algorithm for which the improvement ratio of coalitional deviations can be arbitrarily large. In addition, we design a fully polynomial time approximation scheme (FPTAS), which computes an NE that is a $(1+\varepsilon)$-SE.

We also provide bounds for two other measures of approximate SE, considering the supremum possible improvement of a deviating job and the maximal increase in the cost of non-coalition members, and show that checking whether a specific schedule is an SE is co-NP-complete, which motivates the study of approximate strong equilibria.

Finally, we consider multiple machines. We show that any pure NE on m related machines is a 2-SE. We give improved results for identical machines, and in particular, we show that any pure NE is a 1.32-SE.

1. Introduction

Job scheduling applications have been greatly studied from a game theoretic perspective in recent years, e.g., [26,2,9,10,17]. In a game theoretic setting, the jobs are assumed to play strategically in order to minimize their incurred costs, and might deviate to a different machine in order to decrease their costs. This research agenda is motivated to a

* Corresponding author. Tel.: +972 99602779; fax: +972 9 9568604.
E-mail addresses: lea@math.haifa.ac.il (L. Epstein), mfeldman@cs.huji.ac.il (M. Feldman), tami@idc.ac.il (T. Tamir), lukwit@amu.edu.pl (Ł. Witkowski), mw@amu.edu.pl (M. Witkowski).

0166-218X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.dam.2013.02.035

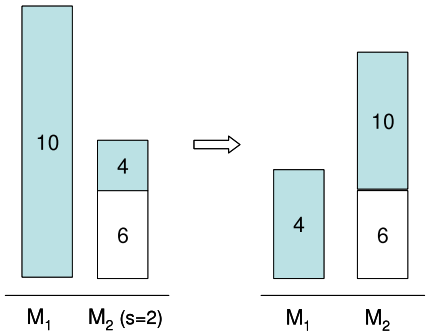

Fig. 1. The left configuration is a Nash equilibrium but is not resilient against coordinated deviations, since the jobs of sizes {10, 4} both profit from deviating to the right configuration.

large extent by Internet applications, which are composed of distributed computer networks that are owned, managed and maintained by diverse entities with potentially competing economic interests [31].

A celebrated notion in game theory is a Nash equilibrium (NE), which is a profile of strategies (one for each player) from which no agent can improve his utility by a unilateral deviation. In such a profile, each agent is best-responding to the strategies played by the other agents. Yet, an NE is not stable against more sophisticated deviations. A stronger solution concept would guarantee stability against coordinated deviations by sets of agents, referred to as "coalitions". A profile that is stable against coalitional deviations is called a *strong equilibrium* (SE) [4]. Specifically, from an SE no coalition can deviate and strictly improve the utility of each one of its members.

The downside of an SE is that such a strong notion of stability rarely exists in games. Yet, several restricted classes of games have been shown to always admit a strong equilibrium [13,23,24,33,29,2], among which is the class of job scheduling games, even under the general unrelated machines setting (where each job can possibly have a different processing time for each machine) [2]. There has also been a considerable amount of work studying the price of anarchy and the price of stability with respect to the SE solution concept in different settings, e.g., [28,2,13,1].

The vast literature on SE has focused on pure strategies and pure deviations, motivated mainly by the fact that the consideration of mixed deviations often results in strong non-existence results. For example, even in settings with identical machines and identical jobs, if mixed deviations are allowed, then an SE rarely exists.

The key observation that motivates our study is that agents would not necessarily engage in deviations that result in very marginal benefits. Indeed, deviations are many times associated with some transition cost (which can be either economical or mental), and this will lead agents to deviate only if the deviation results in a "considerable" improvement. The notion of a "considerable" improvement has been formalized and quantified by Albers [1] in network design settings, and later by Feldman and Tamir [16] in job scheduling settings with identical machines. The key observation established in these works is that even in strategy profiles that are not robust to coalitional deviations, the benefit of the deviating coalition members can be bounded. Depending on the setting at hand, coalitional deviations that can only lead to marginal improvements might not take place. Consequently, it is desired to quantify the benefits that can be generated by coalitional deviations in various settings.

In this paper, we expand the study of Feldman and Tamir [16], who studied the power of coalitional deviations in job scheduling setting of identical machines, to the more general settings of uniformly related machines. Various algorithms have been studied for this setting, carrying different properties with respect to different attributes. While some of these algorithms are extremely desirable from one point of view (e.g., they guarantee a low makespan), they, unfortunately, do not always result in profiles that are SE. Based on the discussion above, it is important to quantify how beneficial coalitional deviations can be from particular profiles that are desirable with respect to other attributes. This is exactly the goal of this paper.

In the uniformly related machines setting, each machine M_i (also called machine i) has a speed s_i associated with it, and each job j has a size $p_j > 0$. The processing time for job j on machine i is p_j/s_i. Given an assignment of jobs into machines, the *load* of a machine is the total size of the jobs assigned to it. The *completion time* of machine i with load L_i is $C_i = L_i/s_i$. We let $C_{\max}$ denote the makespan, that is the maximum value C_i over all machines. The cost of each job is the completion time of the machine it is assigned to. This cost function is commonly used when analyzing systems with negative congestion effect (see [35,32] and references therein). In particular, systems in which jobs are processed in parallel, or when all jobs on a particular machine have the same single pick-up time, or need to share some resource simultaneously. The jobs in our setting are assumed to be owned by rational agents. i.e., each job wishes to minimize its cost. The setting of uniformly related machines is quite involved, and in this paper we make the first steps in its analysis and focus on the case of two machines. In this case, as speeds and job lengths can be scaled, we can assume without loss of generality that the machines' speeds are 1 and $s \geq 1$.

As an example, consider the coalitional deviation depicted in Fig. 1. In this example, three jobs of sizes {10, 6, 4} are assigned to two uniformly related machines with speeds {1, 2}. In the left configuration, $C_1 = 10$ and $C_2 = 5$ (note that in

the figure the load is scaled by the speed). It is easy to verify it is an NE. The coalition consists of the jobs of sizes {10, 4} who coordinately decide to swap locations. In the resulting configuration, at the right, $C_1 = 4$ and $C_2 = 8$. Both jobs decrease their cost, from 10 to 8 and from 5 to 4. The *improvement ratio* of both jobs is $\frac{5}{4}$. Note also that the cost of the job of size 6 (that is not a member of the coalition) increases from 5 to 8.

In the above example, the initial configuration is an arbitrary NE schedule. The notion of approximate strong equilibrium can be also explored with respect to schedules that are obtained by various well-known scheduling algorithms, such as LPT (longest processing time) and LS (list scheduling). LS induces some order on the jobs, and assigns each job (according to that order) to a machine that would minimize the completion time of the job. LPT works in a similar manner, but induces a particular order on the jobs, namely, sorts them in a non-increasing order of their sizes. Any schedule that is obtained by LPT is an NE, which is not necessarily the case for the LS algorithm.

With respect to makespan (i.e., maximum completion time of any machine) minimization, LPT provides a constant approximation ratio for any number of uniformly related machines [11,19,27] (this constant is in $[1.54, 1 + \frac{1}{\sqrt{3}} \approx 1.57735]$ [27]), and of at most $\frac{\sqrt{17}+1}{4} \approx 1.28$ for two machines [20,30]. If the machines are identical, the bounds drop to $\frac{4}{3}$ and $\frac{7}{6}$, respectively [22]. LS provides an approximation ratio of $O(\log m)$ for m machines [8,3], and at most $\phi = \frac{\sqrt{5}+1}{2} \approx 1.618$ for two machines [8,14]. For identical machines, the bounds drop to 2 and $\frac{3}{2}$ [21]. Finally, it is known [8,15,12] that for two unrelated machines, any NE provides a ϕ-approximation to the makespan.

In this paper we study various aspects of approximate strong equilibrium in settings with two uniformly related machines. In particular, we provide bounds for the three measures introduced in [16], related to the improvement ratios of the coalition members and the damage ratios of the non-coalition members. Let Γ be a coalition of agents deviating from a schedule q to a schedule q'. The improvement ratio of a coalition member $j \in \Gamma$, is defined as the ratio between its cost in the original schedule q and its cost in the new one q'. The *minimal improvement ratio* of q' with respect to q is the minimal improvement ratio of any coalition member. The stability of a schedule q is defined by $\mathrm{IR}_{\min}(q)$, which is the maximum over all possible deviation outputs q' of the minimal improvement ratio of q' with respect to q. In other words, there is no coalitional deviation originated from q such that every member of the coalition reduces its cost by a factor greater than $\mathrm{IR}_{\min}(q)$. The $\mathrm{IR}_{\min}$ of a scheduling problem is the supremum value of $\mathrm{IR}_{\min}(q)$ over all valid schedules q. We say that a schedule q is an α-SE if $\mathrm{IR}_{\min}(q)$ is at most α. This notion has been studied in network design games [1] and job scheduling games with identical machines [16].

The identical machines setting was extensively studied in [16]. It was shown that for two identical machines, every NE is an SE (i.e., in the special case of two uniformly related machines where $s = 1$, the value of $\mathrm{IR}_{\min}$ is 1), but for at least three identical machines, the supremum value of $\mathrm{IR}_{\min}$ is at least $\frac{5}{4}$, which is tight for three machines. The 5/4 bound was shown to hold for $m = 4$ (Chen [6]), and very recently, it was shown to hold for every $m \geq 3$ in a working paper by Chen et al. [7]. For multiple identical machines, it was shown [16] that the value of $\mathrm{IR}_{\min}$ is at most 2. For an NE schedule computed using LPT, the tight bounds for $m = 3, 4$ identical machines are $\frac{1}{2} + \frac{\sqrt{6}}{4} \approx 1.1123$ and $\frac{1}{2} + \frac{\sqrt{345}}{30} \approx 1.119$ respectively [16,6], and an upper bound of $\frac{4}{3}$ on the $\mathrm{IR}_{\min}$ is known [16].

We show that for two related machines with speeds 1 and s, every NE is a $(1 + \frac{1}{s} - \frac{1}{s^2})$-SE, and that this bound is tight; i.e., there exists an NE from which a coalition can deviate such that each member improves by a factor of $1 + \frac{1}{s} - \frac{1}{s^2}$ (which is at most 1.25). For schedules obtained from the LPT algorithm we show an upper bound of $1 + \frac{s-1}{2s^2+s}$ (which is smaller than the bound for NE schedules, and can be at most 1.1011). This bound is almost tight. In stark contrast, for schedules obtained from the LS algorithm, the improvement ratio of coalition members can be arbitrarily large for every $s > 1$.

While LPT performs relatively well, this is not the best approximation one can hope for. In particular, in Section 5, we design a poly-time algorithm that for every $\varepsilon > 0$ produces a schedule which is a $(1 + \varepsilon)$-SE (and is an NE).

The two additional measures introduced in [16] are $\mathrm{IR}_{\max}$ and $\mathrm{DR}_{\max}$, defined as follows: (i) $\mathrm{IR}_{\max}(q)$ of a schedule q is the maximum possible improvement ratio among a coalition's members; and (ii) $\mathrm{DR}_{\max}(q)$ of a schedule q is the maximum possible damage ratio of an agent outside a coalition. $\mathrm{IR}_{\max}$ and $\mathrm{DR}_{\max}$ of a scheduling problem are the supremum values of $\mathrm{IR}_{\max}(q)$ and $\mathrm{DR}_{\max}(q)$ over all valid schedules q.

For identical machines, it has been showed in [16] that $\mathrm{IR}_{\max}(q)$ of an NE schedule q can be arbitrarily large, but it is bounded in the interval $[\frac{5}{3}, 2]$ for schedules obtained by LPT. In addition, tight bounds have been provided for the damage ratio measure. In particular, the value of $\mathrm{DR}_{\max}$ is 2 for NE schedules, while this value decreases to $\frac{3}{2}$ for schedules obtained by LPT. Under two related machines we show in this paper that for NE schedules, $\mathrm{IR}_{\max}$ can be arbitrarily large for every $s > 1$, and the damage ratio is bounded by $\frac{s}{s-1}$. For LS schedules, the damage ratio can be arbitrarily large as well.

In Section 6 we show that the problem of checking whether a given NE is an SE is co-NP-complete for every $s > 1$ (for any number of machines $m \geq 2$). The hardness of computing an SE in a setting of related machines is still open. Clearly, if the problem turns out to be hard, our approximation results do not only explain the stability of various schedules to coalitional deviations, but also has computational implications. In particular, the FPTAS we devise computes a $(1+\varepsilon)$-SE in polynomial time. This FPTAS combines the known FPTAS of Horowitz and Sahni [25] for the minimum makespan problem with the local jump-operations technique introduced by Schuurman and Vredeveld in [34].

Finally, in Appendix we consider settings with multiple machines. We extend the upper bound of [16] on the $\mathrm{IR}_{\min}$, and show an upper bound of 2 for related machines. In the Appendix, we provide some preliminary results for multiple *identical*

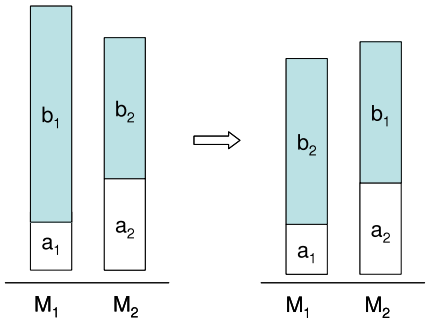

Fig. 2. A general structure of a coalition migrating from the left schedule to the right one.

machines. Specifically, the upper bound is improved to $\frac{33}{25} = 1.32$, and a tight bound of $\frac{5}{4}$ is established for $m = 5$ identical machines.

2. Model and preliminaries

2.1. Resilience to coalitions

A game is denoted by a tuple $G = \langle N, (Q_j), (c_j)\rangle$, where $N = \{1, \ldots, n\}$ is the set of players, Q_j is the finite action space of player $j \in N$, and c_j is the cost function of player j. The joint action space of the players is $Q = \times_{i=1}^{n} Q_i$. For a joint action $q = (q_1, \ldots, q_n) \in Q$, we denote by q_{-j} the actions of players $j' \neq j$, i.e., $q_{-j} = (q_1, \ldots, q_{j-1}, q_{j+1}, \ldots, q_n)$. Similarly, for a set of players Γ we denote by q_Γ and $q_{-\Gamma}$ the actions of players $j \in \Gamma$ and $j \notin \Gamma$, respectively. The cost function of player j maps a joint action $q \in Q$ to a real number, i.e., $c_j : Q \to \mathcal{R}$.

A joint action $q \in Q$ is a *pure Nash equilibrium* (NE) if no player $j \in N$ can benefit from unilaterally deviating from his action to another action, i.e., $\forall j \in N\ \forall a \in Q_j : c_j(q_{-j}, a) \geq c_j(q)$.

A pure joint action of a set of players $\Gamma \subseteq N$ (also called *coalition*) specifies an action for each player in the coalition, i.e., $q'_\Gamma \in \times_{j\in\Gamma} Q_j$. A joint action $q \in Q$ is not resilient to a *pure* deviation of a coalition Γ if there is a pure joint action q'_Γ of Γ such that $c_j(q_{-\Gamma}, q'_\Gamma) < c_j(q)$ for every $j \in \Gamma$ (i.e., the players in the coalition can deviate in such a way that *each* player strictly reduces its cost). In this case we say that the deviation to $q' = (q_{-\Gamma}, q'_\Gamma)$ is a profitable deviation for the coalition Γ. We assume that all the jobs in the coalition Γ deviate (that is, migrate to another machine). Note however that additional jobs might benefit from the joint action of Γ.

A pure joint action $q \in Q$ is *resilient to pure deviations of coalitions* if there is no coalition $\Gamma \subseteq N$ that has a profitable deviation from q.

Definition 1. A pure *strong equilibrium (SE)* is a pure joint action that is resilient to pure deviations of coalitions.

Clearly, a strong equilibrium is a refinement of the notion of Nash equilibrium. In particular, if q is a strong equilibrium, it is resilient to coalitions of size 1, which coincides with the definition of NE.

2.2. Job scheduling on two uniformly related machines

Consider two uniformly related machines, M_1, M_2, with speeds $s_1 = 1$ (slow) and $s_2 = s \geq 1$ (fast). There is a set of jobs $N = \{1, \ldots, n\}$. Each job has a size denoted by p_j. The processing time for job j on machine i is p_j/s_i. A schedule q is an assignment of jobs to machines. Let $L_1(q), L_2(q)$ denote the total load (i.e. sum of p_j) of jobs assigned to the slow and the fast machines, respectively, under assignment q. Let $C_1(q), C_2(q)$ denote the completion times of the machines. It holds that $C_1(q) = L_1(q)$ and $C_2(q) = L_2(q)/s$. When clear in the context, we drop the q from the notation.

A coalition Γ is composed of a set of jobs. Given a schedule q, we denote by b_1 and b_2 the total load of jobs from M_1 and M_2, respectively, that move to the other machine (see Fig. 2), and by a_1 and a_2 the total load of jobs on M_1 and M_2, respectively, that stay on the original machines.

Note that if the original profile, q, is an NE then it must hold that $b_1 > 0$ and $b_2 > 0$, since if a coalition has jobs from only one machine, then every job of the coalition would benefit from moving to the second machine even without other jobs, contradicting the fact that q is an NE.

Let q' denote the post-deviation schedule. We denote the load and the completion time of machine i after the deviation by $L_i(q')$ and $C_i(q')$, respectively. When clear in context, we use L'_i, C'_i. Note that $L'_1 = a_1 + b_2$, $L'_2 = a_2 + b_1$.

2.3. Stability measures

Let q be a schedule that is not resilient to a coalition Γ. The *improvement ratio* of a job $j \in \Gamma$ s.t. $q_j = M_i$ and $q'_j = M_k$ (i.e., a job migrating from machine M_i to machine M_k) is denoted by $\mathrm{IR}^{q,q'}(j) = C_i(q)/C_k(q')$. For any job $j \in \Gamma$, $\mathrm{IR}^{q,q'}(j) > 1$.

The *damage ratio* of a job j s.t. $q_j = q'_j = M_i$ is denoted by $\mathrm{DR}^{q,q'}(j) = C_i(q')/C_i(q)$. For a job $j \in \Gamma$ s.t. $q_j = q'_j = M_i$, it must hold that $C_i(q')/C_i(q) \leq 1$. In contrast, the damage ratio of a job $j \notin \Gamma$ might be greater than 1 (e.g., the job of size 6 in the example depicted in Fig. 1).

If $q_j \neq q'_j$, we say that job j *migrates* in the deviation. Note that, in our terminology, a job can be a member of a profitable deviation even if it does not migrate in the deviation (as long as its cost strictly decreases). Yet, every job that migrates in the deviation is a member of the deviating coalition by definition.

Let $r_{1,2}, r_{2,1}$ denote the improvement ratio of jobs migrating to the fast and slow machine respectively. We have

$$r_{1,2} = \frac{s(a_1 + b_1)}{a_2 + b_1} \quad \text{and} \quad r_{2,1} = \frac{a_2 + b_2}{(a_1 + b_2)s}. \tag{1}$$

The following three measures evaluate how well a configuration approximates SE.

Definition 2. Given schedules q and $q' = (q_{-\Gamma}, q'_{\Gamma})$, the *minimal improvement ratio* of q' with respect to q is $\mathrm{IR}_{\min}(q, q') = \min_{j\in\Gamma} \mathrm{IR}^{q,q'}(j)$. In addition, the *max–min improvement ratio of a schedule* q is $\mathrm{IR}_{\min}(q) = \max_{q'=(q_{-\Gamma},q'_{\Gamma}),\Gamma\subseteq N} \mathrm{IR}_{\min}(q, q')$.

Given schedules q and $q' = (q_{-\Gamma}, q'_{\Gamma})$, the *maximal improvement ratio* of q' with respect to q is $\mathrm{IR}_{\max}(q, q') = \max_{j\in\Gamma} \mathrm{IR}^{q,q'}(j)$. In addition, the *maximal improvement ratio of a schedule* q is $\mathrm{IR}_{\max}(q) = \max_{q'=(q_{-\Gamma},q'_{\Gamma}),\Gamma\subseteq N} \mathrm{IR}_{\max}(q, q')$.

Given schedules q and $q' = (q_{-\Gamma}, q'_{\Gamma})$, the *maximal damage ratio* of q' with respect to q is $\mathrm{DR}_{\max}(q, q') = \max_{j\in N} \mathrm{DR}^{q,q'}(j)$. In addition, the *maximal damage ratio of a schedule* q is $\mathrm{DR}_{\max}(q) = \max_{q'=(q_{-\Gamma},q'_{\Gamma}),\Gamma\subseteq N} \mathrm{DR}_{\max}(q, q')$.

Clearly, for any schedule q on two uniformly related machines, and deviation with improvement ratios $r_{1,2}, r_{2,1}$ as defined in Eq. (1), it holds that $\mathrm{IR}_{\min}(q) = \min\{r_{1,2}, r_{2,1}\}$.

The notion of α-SE [1] is defined in terms of the minimum improvement ratio as follows:

Definition 3. A schedule q is an *α-strong equilibrium (α-SE)* if $\mathrm{IR}_{\min}(q) \leq \alpha$.

2.4. Considered schedules and useful observations

In this paper, we analyze three types of schedules:

- Schedules that are NE.
- Schedules that result from the List Scheduling (LS) algorithm [21]. The LS algorithm is a greedy algorithm that assigns jobs in arbitrary order, each job to a machine that would minimize its completion time, given the current schedule. The schedules produced by LS algorithm are not guaranteed to be NE schedules.
- Schedules that result from the Longest Processing Time (LPT) algorithm [22]. The LPT algorithm is the LS algorithm that assigns the jobs in non-increasing order of their lengths. Schedules that result from (LPT) algorithm are known to be NE schedules [18].

When we consider profitable deviations from these schedules, we refer to them as *NE-originated profitable deviation*, *LS-originated profitable deviation* and *LPT-originated profitable deviation*, respectively.

Next, we provide several useful observations with respect to NE-originated profitable deviation.

The first observation shows that in any NE-originated profitable deviation, if a job migrates to some machine, some other job must migrate out of that machine. This observation holds for any number of machines. Formally:

Observation 4. *Let q be an NE and let $q' = (q_{-\Gamma}, q'_{\Gamma})$ be a profitable deviation. If $q'_j = M_i$ for some $j \in \Gamma$, then $\exists j' \in \Gamma$ s.t. $q_{j'} = M_i$ and $q'_{j'} = M_{i'}$ for some $i' \neq i$.*

This is valid since if job j strictly decreases its cost by migrating to a machine that no other job leaves, it can also profitably migrates unilaterally, contradicting q being an NE.

The next observation limits the NE-schedules in which a coalition might exist to those in which the makespan is determined by the slow machine.

Observation 5. *If a schedule q is an NE but not an SE, then the completion time of the slow machine is larger than the completion time of the fast machine; that is $L_1(q) > \frac{L_2(q)}{s}$. Moreover, $a_2 \geq a_1$ and $b_1 > b_2$.*

Proof. Given that q is not an SE, let Γ be any coalition. Let $r = \min\{r_{1,2}, r_{2,1}\}$. Since q is an NE, there must be at least one job migrating from each machine to the other one (otherwise, a job can migrate alone, contradicting the fact that q is an NE). Therefore, both $r_{1,2}$ and $r_{2,1}$ are greater than 1, and consequently $r > 1$. We have $L'_1 \leq \frac{L_2}{sr}$ and $\frac{L'_2}{s} \leq \frac{L_1}{r}$. Assume without loss of generality that $L_1 + L_2 = L'_1 + L'_2 = 1$. We get

$$1 = L'_1 + L'_2 \leq \frac{1}{r}\left(sL_1 + \frac{L_2}{s}\right) = \frac{1}{r}\left(s(1 - L_2) + \frac{L_2}{s}\right) < s(1 - L_2) + \frac{L_2}{s}.$$

The last inequality follows from $r > 1$. Thus, $L_2(s - \frac{1}{s}) < s - 1$, implying $L_2 < \frac{s}{s+1}$, and $1 - L_2 = L_1 > \frac{1}{s+1}$. Therefore, $L_1 > \frac{L_2}{s}$.

Next, we prove $a_2 \geq a_1$: Using $r_{2,1} = \frac{a_2+b_2}{s(a_1+b_2)} > 1$ implies (using $s \geq 1$) $a_1 + b_2 \leq s(a_1 + b_2) < a_2 + b_2$.

Finally, since $L_1 > \frac{L_2}{s}$ and $L'_1 \leq \frac{L_2}{sr} \leq \frac{L_2}{s}$, we have $L_1 > L'_1$. Since $L_1 = a_1 + b_1$ and $L'_1 = a_1 + b_2$, we get $b_1 > b_2$. □

3. Approximate strong equilibrium by Nash equilibria

Theorem 6. *For any $s > 1$ and a Nash equilibrium configuration q, $\mathrm{IR}_{\min}(q) \leq 1 + \frac{1}{s} - \frac{1}{s^2}$, and this bound is tight.*

Proof. We start with the lower bound for $s > 1$. Consider the following three jobs j_1, j_2, j_3 of the sizes

$$p_1 = s^3 + s^2 - s, \qquad p_2 = s^3 - s^2 \quad \text{and} \quad p_3 = s(s-1)^2(s+1) = s^4 - s^3 - s^2 + s,$$

where the first job is assigned to the first machine, and the other jobs to the second machine. Note that for $s = 2$, this is the instance described in Fig. 1. In this schedule, $C_1 = s^3 + s^2 - s > s^3 - 2s + 1$ and $C_2 = (s^2 + s - 1)(s - 1) = s^3 - 2s + 1$. It is an NE: clearly, no job from M_2 would like to migrate to M_1. Also, moving j_1 to the fast machine would result in $C_2 = s^3 + s^2 - s$, so it is not beneficial for j_1 to move.

Consider now the coalition consisting of jobs j_1 and j_2. Swapping these two jobs would result in $C_1 = s^3 - s^2$ and $C_2 = s^3$. This gives

$$\mathrm{IR}_{\min} = \min\left\{\frac{p_1}{s^3}, \frac{s^3 - 2s + 1}{s^3 - s^2}\right\} = 1 + \frac{1}{s} - \frac{1}{s^2}.$$

To prove an upper bound, consider a schedule q which is an NE, but not an SE. Let Γ be a coalition. Let $r = \min\{r_{1,2}, r_{2,1}\}$ be the minimum improvement ratio.

By Observation 5 (which can be used since q is an NE but not an SE), $C_1 > C_2$. Replace the jobs assigned to M_2 in q by very small jobs ("sand"), and call the new schedule $\hat{q}$. This is still an NE, since jobs still do not benefit from moving unilaterally from the second machine to the first machine, which has a larger completion time, even without any additional jobs. Clearly, any coalition is still valid.

We modify the coalition without changing the value of r. Specifically, the new coalition consists of all jobs that were in the coalition previously, all jobs of the first machine, and an additional total of size a_1 of the small jobs on M_2 that were not in the coalition previously (this is possible since $a_1 \leq a_2$). Let $L_1 = a_1 + b_1$, and $L'_1 = b_2 + a_1$ and $L_2 = b_2 + a_2$. By Observation 5, $L_1 > \frac{L_2}{s}$. Combining with Eq. (1), we get $r = \min\{\frac{sL_1}{L_2 - L'_1 + L_1}, \frac{L_2}{sL'_1}\}$. Since $\hat{q}$ is an NE, we have $\frac{L_2+L_1}{s} \geq L_1$, or $L_2 \geq (s-1)L_1$.

Assume towards contradiction

$$\frac{sL_1}{L_2 - L'_1 + L_1} > \frac{s^2 + s - 1}{s^2} \quad \text{and} \quad \frac{L_2}{sL'_1} > \frac{s^2 + s - 1}{s^2}.$$

We get $s^3 L_1 > (s^2+s-1)(L_2 - L'_1 + L_1)$ and $sL_2 > (s^2+s-1)L'_1$, so $s^3 L_1 > (s^2+s-1)(L_2 - L'_1 + L_1) > (s^2+s-1)(L_2+L_1) - sL_2$. Rewriting this gives $(s^3 - s^2 - s + 1)L_1 > (s^2 - 1)L_2 \geq (s-1)(s^2-1)L_1 = (s^3 - s^2 - s + 1)L_1$, which is a contradiction. □

Note that $1 + \frac{1}{s} - \frac{1}{s^2} \leq 1.25$, and the maximum is obtained for $s = 2$.

Next we show that the supremum improvement ratio is unbounded and provide a tight upper bound for the damage ratio.

Theorem 7. *For any $s \geq 1$, $r > 1$, there exists an NE schedule q on two related machines with speeds $1, s$, for which $\mathrm{IR}_{\max}(q) \geq r$. In addition, for every NE schedule q, the damage ratio is at most $\frac{s}{s-1}$ and this is tight.*

Proof. Let q be an NE schedule that is not an SE. We first show that jobs on M_1 cannot be damaged at all. Formally, $C'_1 < C_1$. This is valid since $C'_1 < C_2$, as some jobs improve by migrating to M_1, and since, by Observation 5, $C_2 < C_1$. Hence, we only need to consider the damage caused to jobs on M_2 that are outside the coalition. Since q is a Nash equilibrium, it is not beneficial for a single job to migrate from M_1 to M_2. Therefore, it is clearly not beneficial for all the jobs on M_1 to migrate together to M_2. Thus, we can bound the completion time of the slow machine, M_1, by $C_1 \leq C_2 + \frac{C_1}{s}$, implying $C_1 \leq \frac{s}{s-1}C_2$. The maximum damage ratio can be bounded by analyzing the case in which all jobs from M_1 migrate to M_2. By the above, it is bounded by

$$\frac{C_2 + \frac{1}{s-1}C_2}{C_2} = \frac{s}{s-1}.$$

Next, we show that the above analysis is tight and in addition, that for any given r, there is a set of jobs and a schedule $\tilde{q}$ for which $\mathrm{IR}_{\max}(\tilde{q}) \geq r$; i.e., the value $\mathrm{IR}_{\max}$ of a schedule can be arbitrarily large. Consider a schedule of three jobs.

$p_1 = 1$, $p_2 = \varepsilon$, and $p_3 = s - 1 - \varepsilon$, where $0 < \varepsilon < \frac{s-1}{sr}$. Assume that the first job is assigned to M_1, and the other jobs to M_2. This schedule is an NE. Consider the coalition consisting of the first and the second jobs. The damage ratio of the third job is $\frac{p_3+p_1}{p_3+p_2} = \frac{s-\varepsilon}{s-1}$. This value tends to $\frac{s}{s-1}$ for small enough ε. The improvement of the second job is $\frac{s-1}{s\varepsilon} > r$, for any $s > 1$ (since $\varepsilon < \frac{s-1}{sr}$). □

Note that $\mathrm{IR}_{\max}$ is unbounded independently of s, while the damage ratio can be arbitrarily large by letting s tend to 1.

4. The performance of LPT and LS algorithms

4.1. Longest Processing Time (LPT)

In this section we consider a subclass of NE, produced by the *Longest Processing Time* (LPT) rule [22]. The LPT rule sorts the jobs in a non-increasing order of their sizes and greedily assigns each job to the machine on which it will be completed first. As shown in [18], every configuration produced by LPT in the related machines model is an NE. In what follows we scale the job sizes so that $L_1 + L_2 = 1$. Let γ be the size of the last job assigned to the slow machine during the LPT process. The following lemma holds for any LPT schedule.

Lemma 8. *It holds that* $L_1 - \frac{L_2}{s} \le \frac{\gamma}{s}$.

Proof. Let λ_1 and λ_2 denote the machine loads at the time of assignment of the last job of the slow machine. By the LPT rule, $\lambda_1 + \gamma \le \frac{\lambda_2+\gamma}{s}$. The final loads satisfy $L_1 = \lambda_1 + \gamma$ and $L_2 \ge \lambda_2$, thus, $L_1 = \lambda_1 + \gamma \le \frac{\lambda_2+\gamma}{s} \le \frac{L_2+\gamma}{s}$. The assertion of the lemma follows. □

In the remainder of the section we will be interested in schedules created by LPT that are not SE schedules. We prove an auxiliary lemma regarding such schedules.

Lemma 9. *Let q be a schedule produced by LPT on two uniformly related machines, that is not an SE. The slow machine must be assigned at least two jobs, and* $L_1 \le \frac{2}{2s+1}$.

Proof. Given that q is not an SE, let Γ be any coalition. Let $r = \min\{r_{1,2}, r_{2,1}\}$. Assume towards contradiction that only a single job, j_1 of size b_1 is assigned by LPT to M_1. By Observation 4 we have that $L'_1 = b_2$. We show that $b_2 < b_1$. This holds since $b_2 = L'_1 \le \frac{L_2}{sr} < \frac{L_1}{r} = \frac{b_1}{r} < b_1$, where the first inequality follows from Eq. (1) and the middle strict inequality follows from Observation 5. By LPT, this implies that only jobs arriving after j_1 might migrate from M_2. Let A be the total size of jobs that arrive before j_1. LPT assigns j_1 to M_1, therefore, $\frac{A+b_1}{s} \ge b_1$. Also, since $L'_1 + L'_2 = L_1 + L_2 = 1$, we have $L'_2 = 1 - b_2$ and $L_1 = b_1$. Combining with Eq. (1), we have $\frac{1-b_2}{s} \le \frac{b_1}{r}$ or $1 - b_2 \le \frac{s}{r} b_1$. Due to the total size of jobs we also have $b_2 \le 1 - A - b_1$. Therefore $sb_1 \le A + b_1 \le 1 - b_2 \le \frac{s}{r} b_1$, implying that $r \le 1$. A contradiction is reached and the first assertion of the lemma follows.

As the slow machine has at least two jobs, and the size of the last (and smallest) such job is γ, we have $L_1 \ge 2\gamma$. By Lemma 8, $\frac{L_2+\gamma}{s} \ge L_1$. Since $L_2 = 1 - L_1$, we get $L_1 \le \frac{(1-L_1)+\frac{L_1}{2}}{s}$, implying $L_1 \le \frac{2}{2s+1}$. □

Theorem 10. *Let q be a schedule produced by LPT on two uniformly related machines. Then,* $\mathrm{IR}_{\min}(q) \le 1 + \frac{s-1}{2s^2+s}$.

Proof. Having $\mathrm{IR}_{\min}(q) \le \frac{L_2}{sL'_1}$ and $\mathrm{IR}_{\min}(q) \le \frac{L_1 s}{L'_2}$, we get

$$\mathrm{IR}_{\min}(q) = \mathrm{IR}_{\min}(q)(L'_1 + L'_2) \le L_1 s + \frac{L_2}{s} = L_1 \left(s - \frac{1}{s}\right) + \frac{1}{s}.$$

Combining with the bound on L_1 from Lemma 9 we get the following bound on $\mathrm{IR}_{\min}(q)$:

$$\mathrm{IR}_{\min}(q) \le \frac{s^2 - 1}{s} \cdot \frac{2}{2s+1} + \frac{1}{s} = 1 + \frac{s-1}{2s^2+s}. \quad \square$$

We note that the above function has a maximum at $s = 1 + \sqrt{1.5} \approx 2.225$, for which the upper bound is 1.10102. This should be contrasted with the bound for NE schedules of $1 + \frac{1}{s} - \frac{1}{s^2}$, for which the upper bound is 1.25.

The above analysis is almost tight, as shown in the following example: Let $s = 2.202$ and consider an instance with job sizes $\{2s - 3 + \varepsilon, 1, 1, 1, 1\}$. An LPT schedule has two jobs of size 1 assigned to M_1, and the rest are assigned to M_2. Consider the coalition in which the two jobs on M_1 swap with the largest job on M_2. The improvement ratio of all jobs is 1.101.

Next, we consider the $\mathrm{IR}_{\max}$ and the damage ratio. We show that unlike the general case, these two measures are both bounded for outputs of LPT. We start with an additional property of such schedules.

Lemma 11. *Let q be a schedule produced by LPT on two uniformly related machines, that is not an SE. Then the set of jobs migrating from the fast machine must contain at least one job assigned earlier than the last job assigned to the slow machine; that is, a job size at least γ.*

Proof. We show that at least one job of the coalition is assigned to the fast machine before the last job is assigned to the slow machine. Assume by contradiction that this is not the case. Let λ_i denote the load on machine i before the assignment of the last job on the slow machine. Thus, $\lambda_2 \le a_2$. Since the migration is beneficial, $\frac{b_1+a_2}{s} < a_1 + b_1$. Since γ is the size of the smallest job assigned to the slow machine, we have $b_1 \ge \gamma$.

By the assignment rule of LPT, $\lambda_1 + \gamma \le \frac{\lambda_2+\gamma}{s}$. However, $L_1 = \lambda_1 + \gamma = a_1 + b_1$, so we have $a_1 + b_1 \le \frac{\lambda_2+\gamma}{s}$. Combining the inequalities, we have $\gamma + a_2 \le b_1 + a_2 < s(a_1 + b_1) \le \lambda_2 + \gamma \le a_2 + \gamma$, which is a contradiction. □

With this we are ready to state the bounds on the two additional measures.

Theorem 12. *For every schedule q produced by LPT,* $\mathrm{IR}_{\max}(q) \le \frac{2s-1}{s}$ *and* $\mathrm{DR}_{\max}(q) \le \frac{2s}{2s-1}$.

Proof. We first note that jobs migrating from the slow machine benefit from the migration and thus $a_2 + b_1 < s(a_1 + b_1)$. By Lemma 8, we have $s(a_1 + b_1) \le b_2 + a_2 + \gamma$, so $a_2 + b_1 \le b_2 + a_2 + \gamma$, i.e., $b_1 \le b_2 + \gamma$.

We have $\mathrm{IR}_{\max}(q) \le \max\{\frac{a_2+b_2}{s(a_1+b_2)}, \frac{s(a_1+b_1)}{a_2+b_1}\}$. First, to show $\frac{a_2+b_2}{s(a_1+b_2)} \le \frac{2s-1}{s}$, we prove $a_2 + b_2 \le (2s-1)(a_1 + b_2)$, or equivalently $a_2 \le (2s-1)a_1 + (2s-2)b_2$. By $a_2 + b_1 < s(a_1 + b_1)$ (which holds as Q is not an SE) we get $a_2 < sa_1 + (s-1)b_1 \le sa_1 + (s-1)(b_2+\gamma)$. Using Lemma 11 we have $b_2 \ge \gamma$, so $a_2 < sa_1 + 2(s-1)b_2 < (2s-1)a_1 + (2s-2)b_2$, since $s \ge 1$.

Next, we bound $\frac{s(a_1+b_1)}{a_2+b_1}$. Using $s(a_1+b_2) < a_2+b_2$ (which holds as Q is not an SE) we have $s(a_2+b_1) = s(1-a_1-b_2) = s - s(a_1+b_2) > s - a_2 - b_2 = s - (1-a_1-b_1) = s - 1 + a_1 + b_1$. Therefore, $\frac{s(a_1+b_1)}{a_2+b_1} \le \frac{s^2(a_1+b_1)}{s-1+a_1+b_1} = \frac{s^2}{(s-1)/(a_1+b_1)+1}$. Using $a_1 + b_1 = L_1 \le \frac{2}{2s+1}$ by Lemma 9, $\frac{s^2}{(s-1)/(a_1+b_1)+1} \le \frac{s^2}{(s-1)(2s+1)/2+1} = \frac{2s^2}{2s^2-s+1}$. The inequality $\frac{2s^2}{2s^2-s+1} \le \frac{2s-1}{s}$ is equivalent to $2s^3 \le 4s^3 - 2s^2 + 2s - 2s^2 + s - 1$, or to $(s-1)(2s^2 - 2s + 1) \ge 0$, which holds for any $s \ge 1$.

As noted in the proof of Theorem 7, the only jobs that may have a damage ratio above 1 are those remaining assigned to the fast machine. We have $\mathrm{DR}_{\max}(q) \le \frac{a_2+b_1}{a_2+b_2}$. It is sufficient to prove $\frac{a_2+b_1}{a_2+b_2} \le \frac{2s}{2s-1}$, or equivalently $(2s-1)b_1 \le a_2 + 2sb_2$. By $s(a_1+b_1) \le b_2+a_2+\gamma$, $a_1 \ge 0$, and $\gamma \le b_2$, we get $sb_1 \le a_2 + 2b_2$. By $b_1 \le b_2 + \gamma \le 2b_2$, we get $(s-1)b_1 \le (2s-2)b_2$. Summing the two inequalities gives $(2s-1)b_1 \le a_2 + 2sb_2$. □

Next, we show that the overall upper bounds of 2 on the $\mathrm{IR}_{\max}$ and the damage ratio cannot be improved. The reason for the relatively high values is that the algorithm scans the input in an online fashion (after sorting it), and given the difference in speed, the machines do not become sufficiently balanced. If a job joins a coalition even if it can improve its cost very slightly, the alternative schedule can be very different from the original one. We construct an example for $s > 3$ that exhibits the situation obtained in the upper bound proof for the $\mathrm{IR}_{\max}$. The slow machine has two jobs roughly of the same size (slightly below s). The fast machine has completion time approximately $2s - 1$, and among other jobs, it has a job of size slightly larger than s. This last job will benefit from moving to the slow machine if its jobs move to the fast machine. The two jobs of the slow machine will indeed benefit from moving if the parameters are chosen appropriately (as their completion times on both machines will be approximately $2s$). Let $s' = \lfloor s \rfloor$, and $\rho = s - s'$ ($0 \le \rho < 1$). We use two parameters: a large value M such that $M > 3s + 1$ and a small value $\varepsilon > 0$ such that $\varepsilon < \frac{1}{M+1}$. Let $s > 3$ and consider an input consisting of jobs of sizes: $s^2 - 2s + s\rho$, $s + M\varepsilon$, s, and $s' + 1$ jobs of size $s - \varepsilon$. Note that $s^2 - 2s + s\rho \ge s + M\varepsilon$ for a sufficiently small ε and $s > 3$. The first job is clearly assigned to the fast machine. The second job is assigned to the fast machine too since $\frac{(s^2-2s+s\rho)+(s+M\varepsilon)}{s} \le s - 1 + \rho + M\varepsilon < s + M\varepsilon$. The third job is assigned to the slow machine since $\frac{(s^2-2s+s\rho)+(s+M\varepsilon)+s}{s} > s + \rho \ge s$. The next s' jobs are assigned to the fast machine since $\frac{(s^2-2s+s\rho)+(s+M\varepsilon)+s'(s-\varepsilon)}{s} = s - 1 + \rho + s' + \frac{M-s'}{s}\varepsilon = 2s - 1 + \frac{M-s'}{s}\varepsilon < s + (s-\varepsilon)$ (since $(\frac{M-s'}{s} + 1)\varepsilon \le (M - s' + 1)\varepsilon < 1$ for $\varepsilon < \frac{1}{M+1}$). Finally, the last job is assigned to the slow machine since $\frac{(s^2-2s+s\rho)+(s+M\varepsilon)+(s'+1)(s-\varepsilon)}{s} = s - 2 + \rho + 1 + s' + 1 + \frac{M-s'-1}{s}\varepsilon > 2s - \varepsilon$. The completion times of the machines are $2s - \varepsilon$ (slow machine) and $2s - 1 + \frac{M-s'}{s}\varepsilon$ (fast machine).

We consider the coalition consisting of the job of size $s + M\varepsilon$ and the two jobs assigned to the slow machine. The new completion time of the fast machine is $2s - 1 + \frac{(M-s')}{s}\varepsilon - 1 - \frac{M\varepsilon}{s} + 2 - \frac{\varepsilon}{s} = 2s - \frac{\varepsilon}{s}(s'+1) < 2s - \varepsilon$, since $s' + 1 = s - \rho + 1 > s$. The new completion time of the slow machine is $s + M\varepsilon < 2s - 1 + \frac{M-s'}{s}\varepsilon$. The improvement ratio of the job of size $s + M\varepsilon$ tends to $\frac{2s-1}{s}$ as ε tends to zero. The overall ratio tends to 2 as s grows. The damage ratio of the first job tends to $\frac{2s}{2s-1}$ as ε tends to zero.

Consider next the following example for small values of s. Let $s = 1 + \tau$, where $\tau < \frac{1}{4}$, and consider the four jobs of sizes $1 + 2\tau + \tau^2$, 1, 1, and $2\tau - \tau^2$. The first job is assigned to the fast machine and the second one is assigned to the slow machine. We have $2 < \frac{1+2\tau+\tau^2+1}{1+\tau}$, so the third job is assigned to the slow machine as well. The fourth job is assigned to the fast machine since $\frac{1+4\tau}{1+\tau} < 2 + 2\tau - \tau^2$. The completion times of the machines are 2 (slow machine) and $\frac{1+4\tau}{1+\tau}$. Consider a coalition consisting of the three first jobs. The new completion times of the machines are $1 + 2\tau + \tau^2 < \frac{1+4\tau}{1+\tau}$

and $\frac{2+2\tau-\tau^2}{1+\tau} < 2$. The damage ratio of the jobs remaining on the fast machine is $\frac{2+2\tau-\tau^2}{1+4\tau}$, which tends to 2 as τ tends to zero.

4.2. List Scheduling

List Scheduling (LS) is a greedy scheduling algorithm, where, similarly to LPT, each job is assigned to the machine on which it will complete earliest [21]. However, unlike LPT, the jobs are assigned in an arbitrary order. Two prominent advantages of the LS algorithm are its suitability to online settings and its scarce memory requirements. Both LS and LPT provide a constant approximation to the optimal makespan on two uniformly related machines [8,14], but they carry different game theoretic properties. First, LS does not necessarily produce an NE. Moreover, as we next show, LS does not provide any approximation to an SE. The following theorems show that already for very small instances (consisting of two or three jobs), both the $\mathrm{IR}_{\min}$ and the $\mathrm{DR}_{\max}$ are not bounded (with the exception of the $\mathrm{DR}_{\max}$ being bounded in the case $s = 1$, that is, for identical machines). The intuitive reason for these properties is that small jobs may be scheduled before large jobs, and (if there are multiple such jobs) they can be either split almost equally between the machines, or they are often assigned to the fast machine. After a huge job arrives, it is much better for a small job which is combined with the huge job by LS to be scheduled on the other machine. In all examples below, this is the status after LS schedules all jobs, and as a result, a relatively small job performs a migration. Thus, the bounds are in fact valid even for coalitions of size 1—when the schedule produced by LS is not even an NE.

Theorem 13. *For any given $s \geq 1$ and $r > 1$, there exists a schedule q produced by LS on two related machines having speeds 1, s, for which $\mathrm{IR}_{\min}(q) > r$.*

Proof. First, assume $s > 1$. Given r, consider an instance consisting of two jobs $p_1 = \varepsilon, p_2 = 1$, where $0 < \varepsilon < \min\{s-1, \frac{1}{rs-1}\}$. Both jobs are assigned to the fast machine by LS. This schedule is not an NE. By migrating, j_1 reduces its cost from $\frac{1+\varepsilon}{s}$ to ε, that is, by a factor of $\frac{1+\varepsilon}{s\varepsilon} > r$.

In the case $s = 1$, given r, consider an instance consisting of three jobs $p_1 = 1, p_2 = 1, p_3 = 2r$. The first two jobs are assigned to different machines by LS. Assume without loss of generality that the third job is assigned together with the first job (the other option is symmetric). This schedule is not an NE. By migrating, j_1 reduces its cost from $2r+1$ to 2, that is, by a factor larger than r. □

Theorem 14. *For any given $s > 1$ and $r > 1$, there exists a schedule q produced by LS on two related machines having speeds 1, s, for which $\mathrm{DR}_{\max} > r$. In the case $s = 1$, $\mathrm{DR}_{\max} = 2$.*

Proof. Given s, r, consider an instance consisting of three jobs $p_1 = \max\{r, s\}, p_2 = 1, p_3 = \frac{s^3 r}{s-1}$. LS assigns j_1 and j_3 to the fast machine, and j_2 to the slow machine (the assignment of j_3 is due to $\frac{p_1+p_3}{s} < 1 + p_3$, which is equivalent to $p_3(s-1) > p_1 - s$, that holds as $p_3(s-1) > rs > \max\{r, s\} = p_1 > p_1 - s$). This schedule is not an NE. By migrating, j_1 reduces its cost from $\frac{p_1+p_3}{s} > \frac{s+s^2 r}{s} = sr + 1$ to $p_1 + p_2 \leq sr + 1$. Moreover, the cost of j_2 is increased by a factor of $p_1 + 1 > \max\{r, s\} \geq r$.

Next, assume $s = 1$. For the lower bound, consider an instance consisting of three jobs $p_1 = 1, p_2 = 1, p_3 = 2$. The first two jobs are assigned to different machines by LS. Assume without loss of generality that the third job is assigned together with the first job. This schedule is not an NE. By migrating, j_1 reduces its cost from 3 to 2. Moreover, the cost of j_2 is increased by a factor of 2. To prove the upper bound, consider a schedule created by LS. We show that any coalition only contains jobs migrating from the more loaded machine to the other. Assume otherwise, and let L_1, L_2, and L'_1, L'_2 denote the loads (which are equal to the completion times) of the two machines before and after the migration, respectively. Then we have $L'_1 < L_2$, and $L'_2 < L_1$, which contradicts $L_1 + L_2 = L'_1 + L'_2$. Thus, the coalition contains jobs migrating from the more loaded machine, M_1, to the less loaded machine, M_2.

Let j_ℓ be the last job assigned to M_1. We show that j_ℓ is not part of the coalition. Let T_1, T_2 be the loads of M_1, M_2 before j_ℓ is assigned. Then $T_1 \leq T_2$. The cost of j_ℓ is $T_1 + p_\ell$ and if it moves its cost will be at least $T_2 + p_\ell$ (maybe more, if it moves with additional jobs, or if M_2 is assigned additional jobs after j_ℓ is assigned). Therefore, the migrating jobs have total size at most T_1. These jobs join jobs of size $L_2 \geq T_2$. Since $T_1 \leq T_2 \leq L_2$, the load of M_2 is increased due to the coalition move by factor at most 2. □

5. Computing a $(1+\varepsilon)$-SE

We describe an FPTAS for the problem, i.e., given any $\varepsilon > 0$, the algorithm produces a schedule q so that $\mathrm{IR}_{\min}(q) \leq (1+\varepsilon)$, in time polynomial in the number of jobs and in $1/\varepsilon$. In addition to approximating the $\mathrm{IR}_{\min}$, the FPTAS simultaneously approximates the makespan within a factor of $1+\varepsilon$.

The FPTAS consists of the following two steps.

1. Apply an FPTAS for the minimum makespan problem; e.g., of Horowitz and Sahni [25] (see also [5]).
2. Turn the resulting schedule into a Nash equilibrium, using local jump-operations, as described by Schuurman and Vredeveld [34].

As defined in [34], a jump operation is performed by a single job migrating from one machine to another. A schedule is *jump-optimal* if no jump decreases the makespan. It is easy to verify that a schedule is jump optimal if and only if it is an NE. Given a schedule, it is possible to perform in polynomial time a sequence of jump operations resulting in a jump optimal schedule, having a makespan not larger than the original schedule [34]. Let q be the output of the above process. Clearly, q has makespan at most $(1+\varepsilon)C_{\max}(opt)$ and is an NE, therefore, the algorithm is an FPTAS.

Claim 15. *The output q of the above FPTAS is a $(1+\varepsilon)$-SE.*

Proof. Let $C_i(q)$ denote the completion times of the machines. By the above, q is an NE, therefore, any coalition has jobs from both machines. Let q' denote the schedule produced by a coalitional deviation. The minimal improvement ratio of a job in the coalition is $\min\{\frac{C_1(q)}{C_2(q')}, \frac{C_2(q)}{C_1(q')}\}$. As q' is a feasible schedule, we know that $\max\{C_1(q'), C_2(q')\} \geq C_{\max}(opt)$ and as both $C_1(q), C_2(q) \leq (1+\varepsilon)C_{\max}(opt)$, we get $\min\{\frac{C_1(q)}{C_2(q')}, \frac{C_2(q)}{C_1(q')}\} \leq (1+\varepsilon)$, as required. □

6. Computational complexity

The following theorem shows that for two uniformly related machines the problem of checking whether a given NE is a strong NE is co-NP-complete. This result should be contrasted with two identical machines, where every NE is also an SE, thus verifying whether a given profile is an SE can be done in linear time.

Theorem 16. *For any $s > 1$, the problem of checking whether a given NE is an SE is co-NP-complete.*

Proof. We use a reduction from PARTITION. Consider an instance of PARTITION, that is, let $\{z_1, z_2, \ldots, z_t\}$ be a set of positive integers, such that $\sum_{i=1}^{t} z_i = 2K$. We assume $K \geq s+1 \geq 2$, which can be obtained by multiplying all integers by $2\lceil s\rceil + 2$. Let

$$\alpha = \frac{2s^2+6s-2}{s(s-1)}, \qquad \beta = \frac{3s(s+1)}{s-1}, \qquad \delta = 1 \quad \text{and} \quad \mu = \frac{s+1}{s-1}.$$

Note that $\alpha > 2$, $\beta = \alpha s + s - 2$, and $\mu > \delta$.

We consider jobs of following sizes.

$$p_i = z_i, \qquad p_{t+1} = \beta K, \qquad p_{t+2} = \alpha K - \delta \quad \text{and} \quad p_{t+3} = K + \mu.$$

We claim that the schedule in which the last two jobs scheduled to the slower machine, and the other jobs to the faster machine, is an NE. The completion time of the faster machine in this schedule is $C_2 = \frac{(\beta+2)K}{s}$. The completion time of the slower machine is $C_1 = (\alpha+1)K + \mu - \delta = \frac{\beta+2}{s}K + \mu - \delta > \frac{\beta+2}{s}K$. Thus, we only need to verify that none of the last two jobs obtains a smaller cost by moving to the faster machine, that is $C_2 + \frac{p_{t+2}}{s} \geq C_1$ and $C_2 + \frac{p_{t+3}}{s} \geq C_1$. Indeed, $\frac{(\beta+2)K}{s} + \frac{\alpha K-\delta}{s} \geq \frac{\beta+2}{s}K + \mu - \delta$ is equivalent to $\alpha K \geq s\mu - s\delta + \delta$, which is equivalent to $\frac{2s^2+6s-2}{s(s-1)}K \geq \frac{s(s+1)}{s-1} - s + 1 = \frac{3s-1}{s-1}$, or $(2s^2+6s-2)K \geq 3s^2 - s$, which clearly holds for $K \geq 2$.

$\frac{(\beta+2)K}{s} + \frac{K+\mu}{s} \geq \frac{\beta+2}{s}K + \mu - \delta$ is equivalent to $K \geq s\mu - s\delta - \mu$, which is equivalent to $K \geq 1$.

Next, we consider a possible coalition. We first note that p_{t+1} would never join a coalition, since even if it would be assigned alone on the slower machine, its completion time would be at least $\beta K \geq C_2$, since $\beta > \frac{s+1}{s-1} > \frac{2}{s-1}$, therefore, $\beta > \frac{\beta+2}{s}$. If p_{t+2} joins a coalition, it would have to join p_{t+1} (and possibly additional jobs) on the faster machine, therefore the completion time would be at least $\frac{\beta K+\alpha K-\delta}{s}$. We show that this is at least C_1. We need to show $\frac{\beta K+\alpha K-\delta}{s} \geq \frac{\beta+2}{s}K + \mu - \delta$, which is equivalent to $(\alpha-2)K \geq s\mu - s\delta + \delta$. Substituting the values of α, μ and δ we get, $\frac{8s-2}{s(s-1)}K \geq \frac{3s-1}{s-1}$, which holds since $K \geq s$.

Next, we consider a structure of a possible coalition. By Observation 4, a coalition must contain at least one job from each machine, therefore, p_{t+3} must switch places with some subset of the first t jobs. Let X denote the sum of these jobs. Thus we have $p_{t+2} + X < C_2$ and $\frac{2K-X+p_{t+3}+p_{t+1}}{s} < C_1$, i.e., $C_2 - \frac{X}{s} + \frac{p_{t+3}}{s} < C_1$.

The first inequality is equivalent to $X < -\alpha K + \delta + \frac{\beta+2}{s}K = K + 1$. The second inequality is equivalent to $X > s(\delta-\mu) + K + \mu = K + s\delta + (1-s)\mu = K - 1$. Note that X must be an integer, since it is the sum of a subset of the first t jobs. Therefore, the only value of X that admits a coalition is K. Thus, the schedule is not a strong NE if and only if a partition exists. □

Acknowledgments

The authors thank an anonymous referee who assisted in simplifying Section 5, and all the referees of this paper for many valuable comments. The second author is partially supported by the Israel Science Foundation (grant number 1219/09), the Leon Recanati Fund of the Jerusalem school of business administration and the Google Inter-university center for Electronic Markets and Auctions.

Appendix. Multiple machines

In this section we consider a more general setting, where there are m parallel machines. Let $F(m)$ denote the value of $\mathrm{IR}_{\min}$ for m identical machines, and let $G(m)$ denote the value of $\mathrm{IR}_{\min}$ for m related machines. We will consider schedules that are arbitrary NE. We say that a machine *participates* in a coalition if at least one job leaves this machine or at least one job migrates to this machine. Let $f(m)$ denote the value of $\mathrm{IR}_{\min}$ for deviations in which the number of participating machines is exactly m. By definition, we have $F(m) = \max_{\ell=1}^{m} f(\ell)$.

Claim 17. *The functions F and G are non-decreasing functions of m.*

Proof. Let $m' < m$ and consider an instance and an NE schedule $\bar{q}$ on m' machines, for which there exists a coalition whose deviation improves all costs of participating jobs by a factor of at least α. Let K denote the makespan. We transform this instance and schedule into an instance and schedule for m machines.

For related machines, add $m - m'$ machines of the same speed as the fastest machine (of indices $m' + 1, \ldots, m$), and let σ be this last speed. Add $m - m'$ jobs of size σK, assigned to machines $m' + 1, \ldots, m$. For identical machines, this results in adding $m - m'$ machines (of speed 1) and $m - m'$ jobs of size K. Denote the new assignment by $\tilde{q}$.

The assignment $\tilde{q}$ is an NE; the completion times of the new machines are too large for jobs of the original instance to prefer a different machine, and the new jobs cannot have a smaller completion time on any machine. The coalition that can be defined for $\bar{q}$ can be defined for $\tilde{q}$ as well. □

We next characterize minimal coalitions. A coalition, for which the value of $\mathrm{IR}_{\min}$ is β, is *minimal* if every coalition consisting of a smaller number of jobs has at least one job that improves by a factor smaller than β.

Claim 18. *Every machine participating in a minimal coalition has at least one job that leaves it and at least one job that migrates to it. For identical machines, every machine participating in the coalition has at least two jobs assigned to it.*

Proof. By Observation 4, a machine that has a job joining it must have a job leaving it. Consider a machine that only has a job leaving it. By removing this job from the coalition, we get a smaller coalition with at least the same improvement ratio. This contradicts minimality.

Consider identical machines, and assume by contradiction that some machine has a single job. This job must migrate, but it already has the smallest possible cost, so moving to a different machine cannot reduce its cost, which is a contradiction. □

Theorem 19. *For uniformly related machines, $\mathrm{IR}_{\min} \leq 2$, that is, $G(m) \leq 2$ for all $m \geq 3$.*

Proof. Consider an NE schedule, and a (minimal) coalition. Recall that L_i and L'_i denote the load of machine i before and after the deviation, respectively. Let $r > 1$ denote the value of the $\mathrm{IR}_{\min}$.

Consider the subset of machines M' which participate in the coalition. Consider machine j in M', let X_j be the size of a coalition job migrating to j (that must exist) and let i be a machine from which it migrates. Since the schedule is an NE, the completion times of the machines before the migration satisfy: $\frac{L_i}{s_i} \leq \frac{L_j + X_j}{s_j}$. However, since after the migration this job has a cost that is smaller by a factor of at least r, we get $\frac{L'_j}{s_j} \leq \frac{L_i}{r \cdot s_i}$. Using $L'_j \geq X_j$ we have $\frac{X_j}{s_j} \leq \frac{L_i}{r \cdot s_i}$. This (combined with the very first inequality) gives $\frac{L_i}{s_i} \leq \frac{L_j}{s_j} + \frac{L_i}{r \cdot s_i}$. Rearranging we get $\frac{L_i}{s_i}(1 - \frac{1}{r}) \leq \frac{L_j}{s_j}$. Using $\frac{L'_j}{s_j} \leq \frac{L_i}{r \cdot s_i}$ gives $\frac{L'_j}{s_j} \leq \frac{L_i}{r \cdot s_i} \leq \frac{L_j}{(r-1)s_j}$, or alternatively, $L'_j \leq \frac{L_j}{r-1}$.

Taking the sum over all machines in M' gives $\sum_{j \in M'} L_j = \sum_{j \in M'} L'_j \leq \frac{1}{r-1} \sum_{j \in M'} L_j$. The first equality holds as other machines do not participate in the coalition and therefore the total size of jobs assigned to machines participating in the coalition remains unchanged. Thus, $r - 1 \leq 1$, which implies $r \leq 2$.

Note that the value $r = 2$ cannot be achieved. In such a case, we have $L'_\ell \leq L_\ell$ for all ℓ. For machines not participating in the coalition this is obvious, and otherwise it follows from $L'_\ell \leq \frac{L_\ell}{r-1} = L_\ell$. If some machine satisfies this with strict inequality, we get $\sum_{\ell=1}^{m} L_\ell = \sum_{\ell=1}^{m} L'_\ell < \sum_{\ell=1}^{m} L_\ell$, which is a contradiction. Thus, if $r = 2$, then $L_\ell = L'_\ell$ for all $1 \leq \ell \leq m$. Consider a machine j which participates in the coalition and has a maximal completion time before the deviation $C_j = \frac{L_j}{s_j}$, and a job that migrates to it from another machine i (by the choice of j, $C_i \leq C_j$). We get $\frac{L'_j}{s_j} \leq \frac{L_i}{r s_i} < \frac{L_i}{s_i}$, where the first inequality holds since a job migrates from j to i, and the second one is due to $r = 2$. Using $\frac{L'_j}{s_j} = \frac{L_j}{s_j} \geq \frac{L_i}{s_i}$, where the equality holds by the assumption $L'_j = L_j$, and the inequality holds by the choice of j as a machine of maximum completion time before the deviation, participating in the coalition. This gives a contradiction. Still, the $\mathrm{IR}_{\min}$ is a supremum value and thus we cannot claim that $\mathrm{IR}_{\min} < 2$ based on this observation. □

A.1. Identical machines

Since in the case of identical machines the load and the completion time of a machine in a given schedule are equal, we use both terms interchangeably. In what follows, we always scale the instance and assume that the minimum completion time of any machine is 1. Since machines that do not participate in the coalition have no effect on the value of the $\mathrm{IR}_{\min}$ for a given coalition, we always assume that all machines participate in the coalition (other machines are ignored, but by Claim 17 the result is valid also for the original number of machines). As before, throughout this section we assume a minimal coalition. Using Claim 18, we can assume that if the machines are identical, then each machine has at least two jobs assigned to it.

Claim 20. *Consider a machine i in an NE schedule that contains $t \geq 2$ jobs. Then its completion time satisfies $C_i \leq 1 + \frac{1}{t-1} \leq 2$.*

Proof. Let π_i denote the size of a minimum sized job assigned to i. We have $C_i \geq t \cdot \pi_i$, which implies $\pi_i \leq \frac{C_i}{t}$. Since the schedule is an NE, moving this job to the least loaded machine is not beneficial, so we have $1 + \pi_i \geq C_i$. Combining the two bounds we get $C_i \leq 1 + \pi_i \leq 1 + \frac{C_i}{t}$. Rearranging gives $C_i \leq \frac{t}{t-1}$. Using the fact that $t \geq 2$ completes the proof. □

We consider a schedule q, and a coalition. We use $K = C_1 \geq C_2 \geq \cdots \geq C_m = 1$ to denote the completion times before the deviation of the coalition (without loss of generality we assume that the machines are sorted by completion time), and let $\kappa_1 \geq \kappa_2 \geq \cdots \geq \kappa_m$ denote the sequence of loads after the deviation (here κ_i is not necessarily the load of machine i in the original ordering of machines). That is, if the machines are sorted again by non-increasing loads after the deviation, then κ_i is the load of the ith machine. Let $r = \mathrm{IR}_{\min}(q)$. In what follows, if a schedule is discussed without mentioning which schedule this is, then the schedule before the deviation is considered. We start with simple upper bounds on the loads on the most loaded machine and the least loaded machine after the deviation.

Claim 21. *$\kappa_m \leq \frac{1}{r}$ and $\kappa_1 \leq \frac{C_1}{r}$.*

Proof. Consider a machine whose resulting load after the migration is κ_1. This machine receives at least one migrating job coming from some machine j. Its new load is therefore at most $\frac{C_j}{r} \leq \frac{C_1}{r}$.

A job migrating from machine m must migrate to a machine whose load becomes no larger than $\frac{C_m}{r}$. Thus, the smallest load after the migration cannot exceed $\frac{C_m}{r} = \frac{1}{r}$. □

We let n_i denote the number of jobs leaving machine i in the coalition. Let $\nu_i = \sum_{j=1}^{i} n_j$ and $\nu_0 = 0$. The motivation for this definition is that it is either the case that many jobs leave the most loaded machines, in which case their loads were not very large, or if only a few jobs are leaving, then machines that do not receive jobs coming from these machines will receive jobs coming from less loaded machines, and thus they must have very small resulting loads. For example, as shown in the previous claim, machines that in the process of deviation receive new jobs coming only from machine 1 could potentially have loads up to $\frac{C_1}{r}$. However, there are at most n_1 such machines. All other machines of the schedule after the deviation will have loads that do not exceed $\frac{C_2}{r}$, as every machine must receive a new job in the process of deviation. This can be generalized as follows.

Claim 22. *If $\nu_i < m$, then $\kappa_{\nu_i+1} \leq \frac{C_{i+1}}{r}$.*

Proof. Consider the $\nu_i + 1 \leq m$ most loaded machines after the deviation. At least one of them does not have a job that migrated from machines $1, \ldots, i$ (since only ν_i jobs migrated from the i most loaded machines). Consider such a machine i'. Since at least one job migrated to i', but no job migrated from machines $1, \ldots, i$, then it has a job that migrated from a machine in $\{i+1, \ldots, m\}$, that is, from a machine that had a load of at most C_{i+1}. The new load of i' is at most $\frac{C_{i+1}}{r}$, and thus the minimum load among the $1, \ldots, \nu_i + 1$ most loaded machines is at most $\frac{C_{i+1}}{r}$. □

Consider a simple example where $m = 7$, $n_1 = 2$, $n_2 = 3$, and $n_3 = 3$. In this case we already know by Claim 21 that after the deviation no load exceeds $\frac{C_1}{r}$, and in particular, each of the two largest loads after the deviation is at most $\frac{C_1}{r}$. By the last claim (Claim 22), each of the three next loads is at most $\frac{C_2}{r}$. Additionally, each further load is at most $\frac{C_3}{r}$, as at most five machines can receive jobs coming from machines 1 and 2. Since we already have an upper bound on the minimum load after the deviation, κ_7, we use this last upper bound only for κ_6, even though $n_3 = 3$. Therefore, the first machine i for which $\nu_i \geq m - 1$ has a special role, and we define it as follows. Let k denote an index such that $\nu_k \geq m - 1$ and $\nu_{k-1} < m - 1$. Note that k must exist since each machine has a migrating job, so $n_i \geq 1$, $\nu_i \geq i$, and $\nu_{m-1} \geq m - 1$. Thus, since $\nu_0 = 0$, $1 \leq k \leq m - 1$. In what follows the intuition is that the worst case scenario is that the migrating jobs were distributed exactly in this way; n_1 jobs (coming from machine 1) to the machines of resulting loads $\kappa_1, \ldots, \kappa_{n_1}$, n_2 jobs (coming from machine 2) to machines $\kappa_{n_1+1}, \ldots, \kappa_{n_1+n_2}$ and so on (until the first $m - 1$ machines of the schedule resulting by the deviation are listed), while the least loaded machine after the deviation received a job coming from machine m (since this job must migrate to some machine).

Corollary 23. *For every j s.t. $\nu_i+1 \le j \le \nu_{i+1}$ for some $0 \le i \le k-2$, it holds that $\kappa_j \le \frac{C_{i+1}}{r}$. For every j s.t. $\nu_{k-1}+1 \le j \le m-1$, it holds that $\kappa_j \le \frac{C_k}{r}$.*

We next find a bound on $\sum_{j=1}^{m} \kappa_j$. We have $\sum_{j=1}^{m} \kappa_j = \sum_{i=0}^{k-2} \sum_{j=\nu_i+1}^{\nu_{i+1}} \kappa_j + \sum_{j=\nu_{k-1}+1}^{m-1} \kappa_j + \kappa_m$. Using Corollary 23, $\sum_{j=\nu_i+1}^{\nu_{i+1}} \kappa_j \le n_{i+1} \frac{C_{i+1}}{r}$ for $0 \le i \le k-2$ (since $\nu_{i+1} - \nu_i = n_{i+1}$) and $\sum_{j=\nu_{k-1}+1}^{m-1} \kappa_j \le (m-1-\nu_{k-1}) \frac{C_k}{r}$. Summing this with $\kappa_m \le \frac{C_m}{r}$ gives

$$\sum_{j=1}^{m} \kappa_j \le \frac{1}{r} \left(\sum_{i=1}^{k-1} n_i C_i + (m-1-\nu_{k-1}) C_k + C_m \right).$$

Using $\sum_{j=1}^{m} C_j = \sum_{j=1}^{m} \kappa_j$ (each being equal to the total size of all jobs) we get

$$r \le \frac{\sum_{i=1}^{k-1} n_i C_i + (m-1-\nu_{k-1}) C_k + C_m}{\sum_{j=1}^{m} C_j}.$$

For machine i, by Claim 20, $C_i \le \frac{n_i}{n_i-1}$, since the number of jobs it contains before the deviation is at least n_i. If $n_i > 2$, then $C_i \le \frac{3}{2}$. Let $\mathcal{A}$ denote the set of machines i that have the following two properties: $C_i > \frac{5}{3}$ and $n_i = 2$. Since every machine that contains at least three jobs in q has a load of at most $\frac{3}{2}$, if $i \in \mathcal{A}$, then it has exactly two jobs before the deviation, and both these jobs migrate. There could be, however, other machines of loads above $\frac{5}{3}$, that obviously have exactly two jobs (before the deviation), but only one migrating job.

Claim 24. *If machine $1 \le i \le m$ has exactly two jobs assigned to it in q, then the size of each job assigned to this machine is at least $C_i - 1$ and at most 1.*

Proof. If $i = m$, then the claim holds trivially as $C_m = 1$. Otherwise, consider a job of size π_i assigned to i in q. Since it does not benefit from moving to machine m, we have $\pi_i + 1 \ge C_i$ or equivalently, $\pi_i \ge C_i - 1$. The size of the other job is $C_i - \pi_i \le 1$. Changing the roles of the jobs gives the claim for both jobs. □

Claim 25. *If $r > 1.25$, then $|\mathcal{A}| \le \frac{m}{3}$.*

Proof. We first show that two jobs coming from the machines of $\mathcal{A}$ cannot migrate to the same machine. Clearly, a pair of jobs coming from the same machine of $\mathcal{A}$ migrate to different machines, otherwise they do not benefit from migrating. Consider two jobs of sizes π_1 and π_2 coming from different machines of $\mathcal{A}$, and let $\ell_1 > \frac{5}{3}$ and $\ell_2 > \frac{5}{3}$ be the loads of their machines before the deviation. If these jobs migrate to one machine then we have $\mathrm{IR}_{\min} \le \frac{\ell_1}{\pi_1+\pi_2} \le \frac{\ell_1}{(\ell_1-1)+(\ell_2-1)} < \frac{5/3}{5/3+5/3-2} \le 1.25$, since by Claim 24, $\pi_i \ge \ell_i - 1$, and since the expression achieves its maximum if ℓ_1, ℓ_2 are minimal. Thus this option is impossible under the condition $r > 1.25$.

If there is a job coming from a machine $i \in \mathcal{A}$ that migrates to another machine $j \in \mathcal{A}$, then swap the contents of machines i and j after the migration. Since all jobs that were assigned to machines in $\mathcal{A}$ belong to the coalition, that is, during the migration process these machines are completely emptied, this is just a matter of renaming the machines, and the swap does not add new jobs to the coalition. As a result, machine i does not belong to $\mathcal{A}$ anymore, as at least one its jobs does not migrate and we remove it from the coalition. Since at least one job is removed from the coalition (a job that now stays on machine i), while the coalition remains non-empty (as it originally consisted of at least two jobs). The value of $\mathrm{IR}_{\min}$ cannot decrease and the new coalition has a smaller number of jobs, which contradicts the minimality of the coalition.

Thus, all jobs of $\mathcal{A}$ migrate to machines that are not in $\mathcal{A}$, each to a different machine, so we have $m - |\mathcal{A}| \ge 2|\mathcal{A}|$. This proves the claim. □

Next, we show a simple property.

Claim 26. *For a fixed value of $m \ge 2$, if $k = m-1$, then $r = 1$.*

Proof. Since $n_i \ge 1$ and $\nu_{m-1} \ge m-1$, $k = m-1$ implies that $n_i = 1$ for $1 \le i \le m-2$, and $\nu_{m-2} = m-2$. We have $r \le \frac{\sum_{i=1}^{m-2} C_i + C_{m-1} + C_m}{\sum_{i=1}^{m-2} C_m} = 1$. □

Therefore, in what follows we consider only the cases $1 \le k \le m-2$. Based on the above claims, we bound the value of $F(m)$. We give an alternative proof to the results of [16,6] for $m = 3, 4$ and extend the analysis for larger values of m.

Theorem 27. *For $3 \le m \le 5$, $F(m) \le 1.25$ and for $m \ge 6$, $F(m) \le 1.32$.*

Proof. Clearly, $f(1) = 1$. The case $m = 2$ is covered by Claim 26, since the only possible value of k is 1.

For the case $m = 3$, the only remaining possible value of k is $k = 1$. In this case $n_1 \geq 2$. We get $r \leq \frac{2C_1+C_3}{C_1+C_2+C_3} \leq \frac{2C_1+1}{C_1+2}$, using $C_2 \geq C_3 = 1$. Using $C_1 \leq 2$ (by Claim 20) gives $r \leq \frac{5}{4}$.

For the case $m = 4$, we consider $k = 1$ and $k = 2$. If $k = 1$, then $n_1 \geq 3$, and $r \leq \frac{3C_1+C_4}{C_1+C_2+C_3+C_4} \leq \frac{3C_1+1}{C_1+3}$, using $C_2 \geq C_3 \geq C_4 = 1$. Since $n_1 \geq 3$, $C_1 \leq \frac{3}{2}$. Thus $r \leq \frac{11}{9}$. If $k = 2$, then $n_1 \leq 2$ and $r \leq \frac{2C_1+C_2+C_4}{C_1+C_2+C_3+C_4} \leq \frac{2C_1+C_2+1}{C_1+C_2+2}$, using $C_3 \geq C_4$. This expression attains its maximum for a maximal value of C_1 and a minimal value of C_2. Using $C_1 \leq 2$ and $C_2 \geq 1$ we get $r \leq \frac{6}{5}$.

For $m = 5$ we need to consider $k = 1, 2, 3$. Assume by contradiction $r > 1.25$. In this case $|\mathcal{A}| \leq \frac{m}{3}$ implies $|\mathcal{A}| \leq 1$.

If $k = 1$, then $n_1 \geq 4$ and $C_1 \leq \frac{4}{3}$. We get a bound of $r \leq \frac{4C_1+C_5}{C_1+C_2+C_3+C_4+C_5} \leq \frac{4C_1+1}{C_1+4} \leq 1.1875$.

If $k = 2$, then there are several options. If $n_1 = 3$, then $C_1 \leq \frac{3}{2}$ and we get the ratio $r \leq \frac{3C_1+C_2+C_5}{C_1+C_2+C_3+C_4+C_5} \leq \frac{3C_1+2}{C_1+4} \leq \frac{13}{11} \approx 1.18$. If $n_1 = 2$, we get $n_2 \geq 2$. Since $|\mathcal{A}| \leq 1$, we have $C_2 \leq \frac{5}{3}$. We get the ratio $r \leq \frac{2C_1+2C_2+C_5}{C_1+C_2+C_3+C_4+C_5} \leq \frac{4+2\cdot\frac{5}{3}+1}{2+\frac{5}{3}+3} = 1.25$. If $n_1 = 1$, then $n_2 \geq 3$ and $C_2 \leq \frac{3}{2}$ and we get the ratio $r \leq \frac{C_1+3C_2+C_5}{C_1+C_2+C_3+C_4+C_5} \leq \frac{4C_2+C_5}{2C_2+3C_5} \leq \frac{7}{6} \approx 1.166$.

If $k = 3$, then $2 \leq n_1 + n_2 \leq 3$. The values n_1, n_2, $4 - n_1 - n_2$ are the numbers 1, 1, 2 in some order.

$$r \leq \frac{n_1C_1 + n_2C_2 + (4 - n_1 - n_2)C_3 + C_5}{C_1 + C_2 + C_3 + C_4 + C_5} \leq \frac{2C_1 + C_2 + C_3 + C_5}{C_1 + C_2 + C_3 + C_4 + C_5} \leq \frac{2C_1 + 3}{C_1 + 4} \leq \frac{7}{6}.$$

Thus we have $F(5) \leq 1.25$.

Consider the case $m \geq 6$, and assume $r > 1.25$. Consider the prefix of machines $1, 2, \ldots, h$ that have loads above $\frac{5}{3}$. Clearly, the set of these machines contains $\mathcal{A}$ (and maybe some other machines). Each such machine satisfies $n_i \leq 2$, and in fact has at most two jobs before the deviation, since otherwise $C_i \leq \frac{3}{2}$. Thus, any machine i in this prefix that is not in $\mathcal{A}$ has $n_i = 1$, while each machine $i \in \mathcal{A}$ has $n_i = 1$.

Let $n'_i = n_i$ for $1 \leq i \leq k-1$ and $n'_k = m - 1 - \nu_{k-1} \geq 1$. Thus $\sum_{i=1}^{k} n'_i = m - 1$. Note that $n'_k = 2$ implies $n_k = 2$, as we have $n_k \leq 2$. The value n'_k can be 2, if $\nu_{k-1} = m - 3$ and $n_k = 2$, and otherwise it is 1. The goal is to limit the number of considered jobs to exactly $m - 1$, as we intend to use bounds on $\kappa_1, \ldots, \kappa_{m-1}$.

Assume first that $k \leq h$, and recall that $k < m$. We bound $\sum_{i=1}^{k-1} n_iC_i + (m - 1 - \nu_{k-1})C_k = \sum_{i=1}^{k} n'_iC_i$ as follows. We have $\sum_{i=1}^{k} n'_iC_i = \sum_{i=1}^{k} C_i + \sum_{i=1}^{k}(n'_i - 1)C_i$, where $0 \leq n'_i - 1 \leq 1$ for $1 \leq i \leq k$. We have $\sum_{i=1}^{k}(n'_i - 1) = \sum_{i=1}^{k} n'_i - k = m - 1 - k$, that is, the number of machines of $\mathcal{A}$ among the first k machines is at least $m - 1 - k$ (if $n_k = 2$ and $n'_k = 1$, then the number of machines of $\mathcal{A}$ is $m - k$). Since $|\mathcal{A}| \leq \frac{m}{3}$, we have $m - k - 1 \leq \frac{m}{3}$, or equivalently $k \geq \frac{2m}{3} - 1$. Since $m \geq 6$, this ensures $m - k - 1 \leq k$ (which is equivalent to $k \geq \frac{m-1}{2}$) since $\frac{2m}{3} - 1 \geq \frac{m-1}{2}$ holds for $m \geq 3$. Since $C_1 \geq \cdots \geq C_k$, $\sum_{i=1}^{k}(n'_i - 1)C_i \leq \sum_{i=1}^{m-1-k} C_i$.

The ratio $\frac{\sum_{i=1}^{m-k-1} C_i + \sum_{i=1}^{k} C_i + C_m}{\sum_{i=1}^{m} C_i} = \frac{\sum_{i=1}^{m-k-1} 2C_i + \sum_{i=m-k}^{k} C_i + C_m}{\sum_{i=1}^{m} C_i}$ is maximized for $C_i = C_1$ for $i \leq m - k - 1$ and $C_i = C_m = 1$ for $i \geq m - k$. Thus we get at most $\frac{2(m-k-1)C_1 + (k+1-(m-k))+1}{(m-k-1)C_1 + k + 1}$. Since $C_1 \leq 2$ we get that this value is at most $\frac{3m-2k-2}{2m-k-1}$. Using $k \geq \frac{2m}{3} - 1$ gives that the last value is at most

$$\frac{3m - \frac{4m}{3} + 2 - 2}{2m - \frac{2m}{3} + 1 - 1} = \frac{5}{4}.$$

If $k > h$, then we use $\sum_{i=1}^{k-1} n_iC_i + (m - 1 - \nu_{k-1})C_k = \sum_{i=1}^{h} n_iC_i + \sum_{h+1}^{k-1} n_iC_i + (m - 1 - \nu_{k-1})C_k$. As before, $\sum_{i=1}^{h} n_iC_i \leq \sum_{i=1}^{|\mathcal{A}|} 2C_i + \sum_{|\mathcal{A}|+1}^{h} C_i$. For every $1 \leq i \leq |\mathcal{A}|$, the maximum ratio is achieved if C_i is maximal, so we use $C_i \leq 2$, and replace $\sum_{i=1}^{|\mathcal{A}|} C_i$ in the expression that we are bounding with $2|\mathcal{A}|$. For every $|\mathcal{A}| + 1 \leq i \leq h$, the maximum value is achieved if C_i is minimal, and we use the simple bound $C_i \geq 1$, and replace $\sum_{|\mathcal{A}|+1}^{h} C_i$ in the expression with $h - |\mathcal{A}|$. For every $h + 1 \leq i \leq k$, let $n'_i = n_i$ for $i \leq k - 1$ and $n'_k = m - 1 - \nu_{k-1}$. For these values of i, if $n'_i \geq 2$, then the maximum value is achieved if C_i is maximal. If $n'_i = 2$, then $C_i \leq \frac{5}{3}$ (as $i > h$), and otherwise, $C_i \leq \frac{n'_i}{n'_i - 1}$. If $n'_i = 1$, then the maximum value is achieved if C_i is minimal, that is, we use $C_i \geq 1$.

Let z_j denote the number of indices $h + 1 \leq i \leq k$ such that $n'_i = j$ (so $\sum_{j=1}^{\infty} z_j = k - h$, and $\sum_{i=h+1}^{k} n'_i = \sum_{j=1}^{\infty} jz_j = m - 1 - \nu_h = m - 1 - (h + |\mathcal{A}|)$). We also use $C_i \geq 1$ for $i > k$, and by substituting every load by the bound for it, we get an upper bound of

$$\frac{4|\mathcal{A}| + h - |\mathcal{A}| + \sum_{j=3}^{\infty} j \cdot z_j \frac{j}{j-1} + 2 \cdot \frac{5}{3}z_2 + z_1 + 1}{2|\mathcal{A}| + h - |\mathcal{A}| + \sum_{j=3}^{\infty} z_j \frac{j}{j-1} + \frac{5}{3}z_2 + z_1 + m - k}.$$

Using $m-1-|\mathcal{A}|-h=\sum_{j=1}^{\infty} jz_j$ and $\sum_{j=1}^{\infty} z_j = k-h$ the upper bound becomes

$$\frac{4|\mathcal{A}|+h-|\mathcal{A}|+2z_2+z_1+\sum_{j=3}^{\infty} jz_j+\sum_{j=3}^{\infty} z_j\frac{j}{j-1}+\frac{4}{3}z_2+1}{2|\mathcal{A}|+h-|\mathcal{A}|+z_2+z_1+\sum_{j=3}^{\infty} z_j+\sum_{j=3}^{\infty}\frac{z_j}{j-1}+\frac{2}{3}z_2+m-k}$$

$$=\frac{4|\mathcal{A}|+h-|\mathcal{A}|+\sum_{j=3}^{\infty} z_j\frac{j}{j-1}+\frac{4}{3}z_2+1+(m-1-|\mathcal{A}|-h)}{2|\mathcal{A}|+h-|\mathcal{A}|+\sum_{j=3}^{\infty}\frac{z_j}{j-1}+\frac{2}{3}z_2+m-k+(k-h)}$$

$$=\frac{2|\mathcal{A}|+\sum_{j=3}^{\infty} z_j\frac{j}{j-1}+\frac{4}{3}z_2+m}{|\mathcal{A}|+\sum_{j=3}^{\infty}\frac{z_j}{j-1}+\frac{2}{3}z_2+m}.$$

Let $\Theta=\sum_{j=2}^{\infty} jz_j$. If $\Theta=0$, since $z_j\geq 0$ for $j\geq 2$, then this means $z_j=0$ for $j\geq 2$, and using $|\mathcal{A}|\leq\frac{m}{3}$ we get a ratio of at most $\frac{5}{4}$. Otherwise, let $\eta_j=\frac{jz_j}{\Theta}$. Clearly, $\eta_j\leq 1$, and $\sum_{j=2}^{\infty}\eta_j=1$. We have $\Theta\leq m-1-|\mathcal{A}|-h\leq m-1-2|\mathcal{A}|$, since $h\geq|\mathcal{A}|$.

We start with proving the next two properties. We show that for $j\geq 3$, we have

$$2|\mathcal{A}|+\frac{\Theta}{j-1}+m\leq 1.32\left(|\mathcal{A}|+\frac{\Theta}{j(j-1)}+m\right)$$

for $j\geq 3$ and that

$$2|\mathcal{A}|+\frac{2}{3}\Theta+m\leq\frac{17}{13}\left(|\mathcal{A}|+\frac{\Theta}{3}+m\right),$$

always holds (where $\frac{17}{13}\approx 1.307$, so this gives $2|\mathcal{A}|+\frac{2}{3}\Theta+m\leq 1.32(|\mathcal{A}|+\frac{\Theta}{3}+m)$).

To prove the first property, we need to show that the next expression is non-positive.

$$\begin{aligned}2|\mathcal{A}|+\frac{\Theta}{j-1}+m-1.32\left(|\mathcal{A}|+\frac{\Theta}{j(j-1)}+m\right)&=0.68|\mathcal{A}|+\frac{\Theta(j-1.32)}{j(j-1)}-0.32m\\&\leq 0.68|\mathcal{A}|+\frac{(m-1-2|\mathcal{A}|)(j-1.32)}{j(j-1)}-0.32m.\end{aligned}$$

For any $j\geq 3$, we have $\frac{(j-1.32)}{j(j-1)}\leq 0.28$, so the expression is at most $0.68|\mathcal{A}|+0.28(m-1-2|\mathcal{A}|)-0.32m=0.12|\mathcal{A}|-0.04m-0.28<0$, since $m\geq 3|\mathcal{A}|$.

To prove the second property, we need to show that the next expression is non-positive.

$$\begin{aligned}2|\mathcal{A}|+\frac{2}{3}\Theta+m-\frac{17}{13}\left(|\mathcal{A}|+\frac{\Theta}{3}+m\right)&=\frac{9}{13}|\mathcal{A}|-\frac{4}{13}m+\frac{3}{13}\Theta\\&\leq\frac{9}{13}|\mathcal{A}|-\frac{4}{13}m+\frac{3}{13}(m-1-2|\mathcal{A}|)=\frac{3}{13}|\mathcal{A}|-\frac{m}{13}-\frac{3}{13}<0.\end{aligned}$$

Both properties are proved. Next, we multiply the first one for every $j\geq 3$ by η_j and we multiply the second inequality by η_2 and get:

$$2\eta_j|\mathcal{A}|+\frac{\Theta\eta_j}{j-1}+m\eta_j\leq 1.32\left(\eta_j|\mathcal{A}|+\frac{\Theta\eta_j}{j(j-1)}+m\eta_j\right)$$

for $j\geq 3$ and

$$2\eta_2|\mathcal{A}|+\frac{2}{3}\Theta\eta_j+m\eta_2\leq 1.32\left(\eta_2|\mathcal{A}|+\frac{\Theta\eta_2}{3}+m\eta_2\right).$$

Using $\Theta\eta_j=jz_j$ and summing over $j\geq 2$ we get

$$2|\mathcal{A}|\sum_{j\geq 2}\eta_j+\sum_{j\geq 3}\frac{jz_j}{j-1}+\frac{4}{3}z_2+m\sum_{j\geq 2}\eta_j\leq 1.32\left(\sum_{j\geq 2}\eta_j|\mathcal{A}|+\sum_{j\geq 3}\frac{jz_j}{j(j-1)}+\frac{2}{3}z_2+m\sum_{j\geq 2}\eta_j\right).$$

Using $\sum_{j\geq 2}\eta_j = 1$, we get

$$2|\mathcal{A}| + \sum_{j\geq 3}\frac{jz_j}{j-1} + \frac{4}{3}z_2 + m \leq 1.32\left(|\mathcal{A}| + \sum_{j\geq 3}\frac{z_j}{j-1} + \frac{2}{3}z_2 + m\right).$$

Thus, the expression that we analyze is at most 1.32, and 1.32 is an upper bound on r. □

References

[1] S. Albers, On the value of coordination in network design, SIAM Journal on Computing 38 (6) (2009) 2273–2302.
[2] N. Andelman, M. Feldman, Y. Mansour, Strong price of anarchy, Games and Economic Behavior 65 (2) (2009) 289–317.
[3] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, O. Waarts, On-line load balancing with applications to machine scheduling and virtual circuit routing, Journal of the ACM 44 (1997) 486–504.
[4] R.J. Aumann, Acceptable points in general cooperative *n*-person games, in: A.W. Tucker, R.D. Luce (Eds.), Contributions to the Theory of Games IV, in: Annals of Mathematics Study, vol. 40, Princeton University Press, 1959, pp. 287–324.
[5] Y. Azar, L. Epstein, Y. Richter, G.J. Woeginger, All-norm approximation algorithms, Journal of Algorithms 52 (2) (2004) 120–133.
[6] B. Chen, Equilibria in load balancing games, Acta Mathematicae Applicatae Sinica (English Series) 25 (4) (2009) 723–736.
[7] B. Chen, S. Li, Y. Zhang, Strong stability of Nash equilibria in load balancing games, in: International Symposium on Combinatorial Optimization, University of Oxford, September, 2012 (submitted for publication).
[8] Y. Cho, S. Sahni, Bounds for list schedules on uniform processors, SIAM Journal on Computing 9 (1) (1980) 91–103.
[9] G. Christodoulou, E. Koutsoupias, A. Nanavati, Coordination mechanisms, Theoretical Computer Science 410 (36) (2009) 3327–3336.
[10] A. Czumaj, B. Vöcking, Tight bounds for worst-case equilibria, ACM Transactions on Algorithms 3 (1) (2007).
[11] G. Dobson, Scheduling independent tasks on uniform processors, SIAM Journal on Computing 13 (4) (1984) 705–716.
[12] L. Epstein, Equilibria for two parallel links: the strong price of anarchy versus the price of anarchy, Acta Informatica 47 (7–8) (2010) 375–389.
[13] A. Epstein, M. Feldman, Y. Mansour, Strong equilibrium in cost sharing connection games, Games and Economic Behavior 67 (1) (2009) 51–68.
[14] L. Epstein, J. Noga, S.S. Seiden, J. Sgall, G.J. Woeginger, Randomized online scheduling on two uniform machines, Journal of Scheduling 4 (2) (2001) 71–92.
[15] R. Feldmann, M. Gairing, T. Lücking, B. Monien, M. Rode, Nashification and the coordination ratio for a selfish routing game, in: ICALP, 2003, pp. 514–526.
[16] M. Feldman, T. Tamir, Approximate strong equilibrium in job scheduling games, Journal of Artificial Intelligence Research 36 (2009) 387–414.
[17] A. Fiat, H. Kaplan, M. Levy, S. Olonetsky, Strong price of anarchy for machine load balancing, in: ICALP, 2007, pp. 583–594.
[18] D. Fotakis, S.C. Kontogiannis, E. Koutsoupias, M. Mavronicolas, P.G. Spirakis, The structure and complexity of Nash equilibria for a selfish routing game, Theoretical Computer Science 410 (36) (2009) 3305–3326.
[19] D.K. Friesen, Tighter bounds for LPT scheduling on uniform processors, SIAM Journal on Computing 16 (3) (1987) 554–560.
[20] T. Gonzalez, O.H. Ibarra, S. Sahni, Bounds for LPT schedules on uniform processors, SIAM Journal on Computing 6 (1) (1977) 155–166.
[21] R. Graham, Bounds for certain multiprocessing anomalies, Bell System Technical Journal 45 (1966) 1563–1581.
[22] R. Graham, Bounds on multiprocessing timing anomalies, SIAM Journal on Applied Mathematics 17 (1969) 263–269.
[23] R. Holzman, N. Law-Yone, Strong equilibrium in congestion games, Games and Economic Behavior 21 (1997) 85–101.
[24] R. Holzman, N. Law-Yone, Network structure and strong equilibrium in route selection games, Mathematical Social Sciences 46 (2003) 193–205.
[25] E. Horowitz, S. Sahni, Exact and approximate algorithms for scheduling non-identical processors, Journal of the Association for Computing Machinery 23 (1976) 317–327.
[26] E. Koutsoupias, C.H. Papadimitriou, Worst-case equilibria. in: STACS, 1999, pp. 404–413.
[27] A. Kovács, New approximation bounds for LPT scheduling, Algorithmica 57 (2) (2010) 413–433.
[28] S. Leonardi, P. Sankowski, Network formation games with local coalitions, in: PODC, 2007, pp. 299–305.
[29] I. Milchtaich, Crowding games are sequentially solvable, International Journal of Game Theory 27 (1998) 501–509.
[30] P. Mireault, J.B. Orlin, R.V. Vohra, A parametric worst case analysis of the LPT heuristic for two uniform machines, Operations Research 45 (1997) 116–125.
[31] C.H. Papadimitriou, Algorithms, games, and the Internet, in: STOC, 2001, pp. 749–753.
[32] T. Roughgarden, E. Tardos, How bad is selfish routing? Journal of the ACM 49 (2) (2002) 236–259.
[33] O. Rozenfeld, M. Tennenholtz, Strong and correlated strong equilibria in monotone congestion games, in: WINE, 2006, pp. 74–86.
[34] P. Schuurman, T. Vredeveld, Performance guarantees of local search for multiprocessor scheduling, INFORMS Journal on Computing 19 (1) (2007) 52–63.
[35] B. Vöcking, Chapter 20: selfish load balancing, in: N. Nisan, T. Roughgarden, E. Tardos, V. Vazirani (Eds.), Algorithmic Game Theory, Cambridge University Press, 2007.