Operations Research

Conflicting Congestion Effects in Resource Allocation Games

Michal Feldman, Tami Tamir,

To cite this article:
Michal Feldman, Tami Tamir, (2012) Conflicting Congestion Effects in Resource Allocation Games. Operations Research 60(3):529-540. https://doi.org/10.1287/opre.1120.1051

OPERATIONS RESEARCH
Vol. 60, No. 3, May–June 2012, pp. 529–540
ISSN 0030-364X (print) | ISSN 1526-5463 (online)

http://dx.doi.org/10.1287/opre.1120.1051
© 2012 INFORMS

Conflicting Congestion Effects in Resource Allocation Games

Michal Feldman
School of Business Administration and Center for the Study of Rationality, Hebrew University of Jerusalem, Jerusalem 91905, Israel, mfeldman@huji.ac.il

Tami Tamir
School of Computer Science, The Interdisciplinary Center, Herzliya, Herzliya 46150, Israel, tami@idc.ac.il

We study strategic resource allocation settings, where jobs correspond to self-interested players who choose resources with the objective of minimizing their individual cost. Our framework departs from the existing game-theoretic models mainly in assuming conflicting congestion effects, but also in assuming an unlimited supply of resources. In our model, a job's cost is composed of both its resource's load (which increases with congestion) and its share in the resource's activation cost (which decreases with congestion). We provide results for a job-scheduling setting with heterogeneous jobs and identical machines.

We show that if the resource's activation cost is shared equally among its users, a pure Nash equilibrium (NE) might not exist. In contrast, the *proportional* sharing rule induces a game that admits a pure NE, which can also be computed in polynomial time. As part of the algorithm's analysis, we establish a new, nontrivial property of schedules obtained by the longest processing time algorithm. We also observe that, unlike in congestion games, best-response dynamics (BRD) are not guaranteed to converge to a Nash equilibrium. Finally, we measure the inefficiency of equilibria with respect to the minimax objective function, and prove that there is no universal bound for the worst-case inefficiency (as quantified by the "price of anarchy" measure). However, the best-case inefficiency (quantified by the "price of stability" measure) is bounded by 5/4, and this is tight. These results add another layer to the growing literature on the price of anarchy and stability, which studies the extent to which selfish behavior affects system efficiency.

Subject classifications: congestion games; cost sharing games; job scheduling; potential games; price of anarchy; price of stability; equilibrium existence.
Area of review: Optimization.
History: Received November 2009; revisions received March 2011, September 2011, October 2011; accepted November 2011.

1. Introduction

1.1. Background and Motivation

In resource allocation applications, tasks are assigned to resources to be performed. For example, in job-scheduling applications, jobs are assigned to servers to be processed; similarly, in communication or transportation networks, traffic is assigned to network links to be routed. These settings raise many interesting combinatorial optimization problems, but because they often consist of multiple strategic users, whose individual payoff is affected by the decisions made by others, game theory has become an essential tool in their analysis.

Indeed, the analysis of many operations research applications (such as job scheduling, supply chain management, and queuing networks) has applied game-theoretic concepts and tools. Following the emergence of the Internet, there has been an explosion of studies employing game-theoretic analysis to explore Internet applications, such as routing in computer networks, and the creation and design of Internet-like networks.

In this paper we apply noncooperative game theory to resource allocation applications and study it in job-scheduling terminology. At the heart of the game-theoretic view is the assumption that the players have *strategic* considerations and they act to minimize their own cost, rather than to optimize any global objective. In job-scheduling settings, this would mean that the jobs[1] *choose* a resource instead of being assigned to one by a central designer. The focus in game theory is on the *stable* outcomes of a given setting, or the *equilibrium* points. A Nash equilibrium (NE) is a profile of the users' strategies such that no user can decrease its cost by a unilateral deviation from its current strategy (given that the strategies of the remaining users do not change).

Because the users' cost functions lead them in their decisions, the structure of the cost function is a key component of any game-theoretic treatment of the problem. The literature is divided into two main approaches with respect to the cost function. The first class of models (e.g., routing and job scheduling) emphasizes the negative congestion effect, assuming that a resource's cost increases in the load. The second class (e.g., network creation games), in contrast, emphasizes the positive congestion effects, assuming some fixed *activation cost* that should be covered and divided between the users. Although the first class ignores the positive congestion effects and the second ignores the negative congestion effects, both effects take place in practice. On the one hand, a heavy-loaded resource might be less attractive due to negative congestion effects; on the other hand, a new resource might be too costly to activate. In almost any resource allocation setting, the consideration of these two conflicting congestion effects is a more realistic model of real-life situations, where activating a new "resource" is costly, but at the same time relieves the congestion burden.

A setting where this trade-off naturally occurs is the creation of unregulated networks such as the Internet. Autonomous systems (ASs), in their peering and communication agreements, need to take into account both the hardware cost (i.e., the total cost of laying down the edges) and the quality of service experienced by their customers (i.e., the latency experienced by their traffic). Clearly, laying down an edge is costly, but may, at the same time, pay off in quality of service improvements. While each AS would act to minimize its own cost, from a social perspective it is reasonable to desire that the outcome should be such that no individual would incur too high of a cost. This is a paradigmatic example of the Rawlsian principle (Rawls 1971), which seeks to improve the situation of the weakest member in the society.

It is well known that decentralized decision making, as manifested in equilibrium behavior, may lead to suboptimal solutions from the point of view of society as a whole. In some cases, it may be possible to alleviate the inefficiency through a well-designed incentive scheme (this is the approach taken by, e.g., Parlakturk and Kumar 2004, which proposes a scheduling rule for self-interested routing that achieves performance comparable to the first-best solution). However, in many cases the system rules are already in place, and quantifying the inefficiency that arises in equilibrium behavior is a major challenge (see, e.g., Cominetti et al. 2009). In this paper we quantify the inefficiency using the *price of anarchy* (Papadimitriou 2001, Koutsoupias and Papadimitriou 2009) and *price of stability* (Correa et al. 2004, Anshelevich et al. 2004) measures.

We consider a job-scheduling setting with n jobs of variable lengths, and where the cost of an individual job is composed of (i) the load on the job's chosen machine and (ii) the job's share in the activation cost of the chosen machine.[2] In particular, each machine is associated with some fixed activation cost that should be jointly incurred by the set of jobs using it. Our primary focus is on the *proportional* sharing rule, where the resource's cost is shared among its users in proportion to their size, but we also provide some analysis with respect to the *uniform* sharing rule, where the resource's cost is shared uniformly among its users.

In addition to the consideration of conflicting congestion effects, our model departs from standard job-scheduling models in the assumption about the supply of resources. Although most of the models assume the existence of an a priori fixed set of resources, in many practical settings a user controlling a job has the opportunity to utilize a new resource at its own cost. For example, a user might be able to purchase a dedicated server for his job if he is willing to cover its cost. Similarly, a service provider might wish to invest in new communication lines in order to reduce the transmission delay of its traffic. Consequently, we consider settings with no a priori limit on the number of resources.

Even in our simple model, many interesting phenomena arise. Indeed, the induced game is qualitatively different from earlier models that consider only either positive or negative congestion effects. Because the combination of these effects is so prevalent in resource allocation settings, we believe this work will inspire future work on conflicting congestion effects in more complex settings.

1.2. Our Results

We provide results with respect to the following four themes:

Equilibrium existence (§3): In contrast to previous job-scheduling models, our game in its general form does not comply with the family of *potential games* (or congestion games), for which an NE in pure strategies is known to exist (Rosenthal 1973, Monderer and Shapley 1996) (note that the game always admits an NE in mixed strategies by Nash's theorem 1951 because it is finite). We find that under the uniform sharing rule, a pure NE is not guaranteed to exist. In contrast, the proportional sharing rule, apart from being natural, induces a game that always admits a pure NE.

Because the induced game is not a potential game, we could not establish existence by the standard potential function technique. Instead, we developed two different proofs, each with its own advantage. The proof that is deferred to the appendix is simpler, but not computational. The proof that appears in the body of the paper is technically more involved, but has two main advantages; namely, it runs in polynomial time, and it establishes en route a new property of the LPT algorithm, which may be of interest independently of its game-theoretic interpretation.

Equilibrium inefficiency (§5): The PoA is the worst-case inefficiency of a Nash equilibrium and is defined as the ratio between the social cost of the worst NE and the optimal solution. The PoS measures the best-case inefficiency of a Nash equilibrium and is defined as the ratio between the best NE and the optimal solution. The PoA and PoS measures should be quantified according to a well-defined social cost function. Here, we consider the egalitarian function, also known as *minimax* (Hakimi 1965); i.e., we wish to minimize the cost of the job with the highest cost.

We provide an analysis of the PoA with respect to the proportional sharing rule, as a function of the ratio between the longest job and the activation cost. The PoA is 1 if this ratio is larger than 1. On the other hand, the PoA can be arbitrarily high as this ratio decreases. This result indicates that the loss due to strategic behavior can be very severe. In contrast, we show that the PoS is bounded by 5/4, and this is tight. Interestingly, all of the lower bounds are obtained under the special case of unit-length jobs (for which the two sharing rules coincide).

Computational complexity (§3): Whereas computing a NE may be hard even for potential games, it is computationally feasible for models that are closely related to our model. For example, if a user's cost is the load of its chosen machine, the longest processing time (LPT) algorithm always results in an NE (Fotakis et al. 2002). In our model, LPT is not well defined because the number of machines is not fixed. However, our existence result is constructive and shows that an NE can be computed in polynomial time. We also show, however, that calculating the best NE is an NP-hard problem.

Convergence to equilibrium (§4): It is known that best-response dynamics (BRD) always converge to a pure Nash equilibrium in congestion games (Monderer and Shapley 1996) and in job-scheduling games that do not take activation costs into account (Even-Dar and Mansour 2005). As we show, however, in the setting considered in this paper, under the proportional sharing rule BRD do not always converge to pure NE, although one is guaranteed to exist. Only a few natural games possess this property. The fact that BRD might cycle also implies that our game is not a potential game.

The results for our model are summarized in Table 1, where they are also contrasted with the results obtained with respect to the mini-sum objective (Chen and Gürel 2012). It is interesting to note that whereas the PoA is unbounded in both cases, and is given by the same expression, the PoS is unbounded with respect to the minisum objective, but is bounded by a small constant with respect to the minimax objective. In addition, whereas in our case all of the lower bounds are obtained under the special case of unit-length jobs, the lower bound obtained by Chen and Gürel (2012) employs heterogeneous jobs.

Table 1. Summary of results for our model.

	Uniform sharing	Proportional sharing	Unit length
NE existence [$*$]	NO	YES	YES
BRD convergence [$*$]	N/A	No	YES
Minimax objective [$*$]			
PoA	N/A	$\frac{1+\alpha}{2\sqrt{\alpha}}$	$\frac{1+\alpha}{2\sqrt{\alpha}}$
PoS	N/A	5/4	5/4
Minisum objective (Chen and Gürel 2012)			
PoA	N/A	$\frac{1+\alpha}{2\sqrt{\alpha}}$	$\frac{1+\alpha}{2\sqrt{\alpha}}$
PoS	N/A	UB: $\frac{\rho+1}{2}$ LB: $\frac{1}{3}\sqrt[3]{\rho}$	UB: $\frac{\rho+1}{2}$

Notes. $\alpha = B/p_{\max}$, $\rho = B/p_{\min}$, where B is the activation cost, $p_{\min}$, $p_{\max}$ are the lengths of the shortest and the longest job, respectively. [$*$]: our results.

1.3. Literature Review

A lot of research has been conducted in the analysis of job-scheduling applications using a game-theoretic approach. One branch emphasizes the *mechanism design* approach and seeks *truthful* mechanisms (see the seminal paper by Nisan and Ronen 2001 and the survey by Lavi 2007). The current work belongs to the other branch, which treats the jobs as players whose strategy consists of deciding which machine to choose. The questions that are commonly analyzed under this approach are Nash existence, Nash quality, and convergence of best-response dynamics. The different models that have been studied are a highly diverse group, including, among others, the machine model (e.g., identical/related/unrelated/restricted), the individual cost model (e.g., makespan, completion time), the solution concept (e.g., pure NE, mixed NE, strong equilibrium), the social objective (e.g., minisum/minimax), and more.

The main Nash quality measures that have been featured in the literature are the PoA (Koutsoupias and Papadimitriou 2009, Papadimitriou 2001) and the PoS (Correa et al. 2004, Anshelevich et al. 2004). These metrics have been studied in a variety of applications, such as selfish routing (see Roughgarden 2007 and references therein, Cominetti et al. 2009, Parlakturk and Kumar 2004), network formation (Fabrikant et al. 2003, Albers et al. 2006, Anshelevich et al. 2004) and facility location (Vetta 2002).

For an excellent recent survey on the quality of equilibrium, with respect to the makespan objective, in job-scheduling settings on identical machines and related machines, we refer the reader to Vöcking (2007), Koutsoupias and Papadimitriou (2009), and Czumaj and Vöcking (2002). Specifically, it has been shown that the PoA is bounded by approximately 2 with respect to pure NE, whereas the PoA with respect to mixed NE can be as large as $O(\log m/\log\log m)$, which is also the PoA on related machines with respect to pure NE (where m is the number of machines). When considering a model of both related machines and mixed strategies, the PoA is $O(\log m/\log\log\log m)$. It can be quite easily shown that the PoA in an unrelated machines setting is unbounded. However, it has been shown that the PoA with respect to the strong equilibrium solution concept is (tightly) bounded by m, whereas the PoS is 1 (Andelman et al. 2009). A lower bound for related machines has been studied by Fiat et al. (2007).

Perhaps the closest work to the current paper is a follow-up paper by Chen and Gürel (2012), who study the exact same cost model proposed here (including both negative and positive congestion effects). This work complements our work very nicely, by providing parametric bounds on the PoA and PoS with respect to the mini-sum objective function. The inefficiency of a Nash equilibrium for this mini-sum objective (with no activation costs) is considered also in Correa and Queyranne (2010).

In addition to job-scheduling applications, a game-theoretic analysis has been applied to a large variety of other applications, with an emphasis on the study of equilibrium existence and quality. Some examples are network-routing settings (Correa et al. 2007, 2004; Bonifaci et al. 2010; Cominetti et al. 2009, Awerbuch et al. 2005), network-creation games (Fabrikant et al. 2003, Albers et al. 2006, Anshelevich et al. 2004, Epstein et al. 2007), and cost-sharing games (see Jain and Mahdian 2007 and references therein). A somewhat different game arises in the analysis of supply chain management, which focuses on the mechanism design approach (Cachon and Lariviere 1999) and on equilibrium existence and inefficiency (Perakis 2007, Perakis and Roels 2007, Bernstein and Federgruen 2001). For a survey on a game-theoretic treatment of supply chain management, see Cachon and Netessine (2004). Cooperative game theory is extensively used in the analysis of these applications.

Sharing of joint costs among heterogeneous players is a common problem for which a large number of sharing rules have been proposed (Moulin and Shenker 1992, 2001; Herzog et al. 1997). In a departure from the above literature, which emphasizes the efficiency and fairness properties associated with the different sharing rules, our focus is on the equilibrium existence and the inefficiency obtained due to selfish behavior associated with the different sharing rules.

2. Model and Preliminaries

2.1. Job Scheduling on Identical Machines

Although our setting models many resource allocation problems, we present it using the terminology of job-scheduling for simplicity of presentation. A *job-scheduling setting* with identical machines is characterized by a set of machines $M = \{M_1, M_2, \ldots\}$, and a set of jobs $N = \{1, \ldots, n\}$, where a job $j \in N$ has a length (i.e., processing time) p_j. The set of possible assignments of jobs into machines is denoted by S. An assignment method produces an assignment $s = (s_1, \ldots, s_n) \in S$ of jobs into machines, where $s_j \in M$ denotes the machine that job j is assigned to. The assignment is referred to as a schedule, profile, joint action, or configuration (we use all of these terms interchangeably). The load of a machine M_i in a schedule s is the sum of the processing times of the jobs assigned to M_i, that is, $L_i(s) = \sum_{j: s_j = M_i} p_j$. The *makespan* of a schedule is the load on the most loaded machine; we denote it $L_{\max}(s) = \max_i L_i(s)$.

Given a job-scheduling setting and an activation cost B, a *job-scheduling game* is induced, where the set of players corresponds to the set of individual jobs, and the action space S_j of each player j is the set of individual machines. The cost function of job j in a given schedule is composed of the load on the job's machine and the job's share in the machine's activation cost; i.e., given a profile s in which $s_j = M_i$, the cost of job j is given by:

$$c_j(s) = L_i(s) + b_j(s),$$

where $L_i(s)$ is the total load on machine M_i in s, and $b_j(s)$ is j's share in the activation cost of M_i.

A few notes on the cost function are in order. The load-related component of the cost function (i.e., total load on the chosen machine) characterizes systems in which jobs are processed in parallel, or need to use a shared resource simultaneously, or share the same pickup time; thus, there is no significance to the order of the jobs. These problems have been widely studied in recent years from a game-theoretic perspective by Koutsoupias and Papadimitriou (2009), Czumaj and Vöcking (2002), Fiat et al. (2007), and Feldman and Tamir (2009).

The activation cost component in the cost function depends on the *sharing rule*. Under the *uniform* sharing rule, all the jobs assigned to a particular resource share its cost uniformly, i.e., the cost of job j such that $s_j = M_i$ is $b_j(s) = B/(|\{k: s_k = M_i\}|)$. Under the *proportional* sharing rule, the jobs assigned to a particular resource share its cost proportionally to their lengths, i.e., the cost of job j such that $s_j = M_i$ is $b_j(s) = (p_j B)/L_i(s)$.

As an example, consider an instance with activation cost $B = 12$ and two jobs of lengths 1 and 2, respectively. If both jobs are assigned to the same machine, then their cost under the uniform sharing rule would be $c_1(s) = c_2(s) = 3 + 12/2 = 9$, whereas their respective costs under the proportional rule would be $c_1(s) = 3 + 12/3 = 7$, $c_2(s) = 3 + 2 \cdot 12/3 = 11$.

An assignment $s \in S$ is a *pure Nash equilibrium* (NE) if no job $j \in N$ can benefit from unilaterally deviating from its machine to another machine; i.e., for every $j \in N$ and every $s_j \in M$ it holds that $c_j(s_{-j}, s_j) \geqslant c_j(s)$.

The social cost function of schedule s, denoted $g(s)$, is the highest cost among all the jobs; i.e., $g(s) = \max_j c_j(s)$. We denote by OPT the optimal solution; i.e., $OPT = \min_{s \in S} g(s)$. With this we are ready to define the price of anarchy and the price of stability.

Definition 1. Let $\mathcal{G}$ be a family of games, and let $G \in \mathcal{G}$ be some game in this family. Let $\Phi(G)$ be the set of Nash equilibria of the game G. If $\Phi(G) \neq \varnothing$:

- the *price of anarchy* of the game G is the ratio between the *maximal* cost of a Nash equilibrium and the social optimum of G:

$$PoA(G) = \max_{s \in \Phi(G)} g(s)/OPT(G),$$

and the *price of anarchy* of the family of games $\mathcal{G}$ is

$$PoA(\mathcal{G}) = \operatorname{Sup}_{G \in \mathcal{G}} PoA(G).$$

• the *price of stability* of the game G is the ratio between the *minimal* cost of a Nash equilibrium and the social optimum of G:

$$PoS(G) = \min_{s \in \Phi(G)} g(s)/OPT(G),$$

and the *price of stability* of the family of games $\mathcal{G}$ is:

$$PoS(\mathcal{G}) = \operatorname{Sup}_{G \in \mathcal{G}} PoS(G).$$

2.2. Proportional Sharing Rule—Useful Observations

We present several claims and observations that provide some intuition regarding the behavior of jobs under the proportional sharing rule. These observations shall be used repeatedly in the sequel.

We first specify the condition under which a job is better off migrating from one machine to another. We distinguish between two types of migrations. In a load-increasing migration, the resulting load on the target machine is larger than the load on the original machine, whereas in a load-decreasing migration, the resulting load on the target machine is lower than the load on the original machine. Roughly speaking, a load-increasing (decreasing) migration might be beneficial if the activation cost is relatively low (high). The following assertion, which follows from simple arithmetic, specifies the exact threshold for which each type of migration is beneficial. Clearly, if the target machine has the same load as the source machine, then the job's cost does not change.

Claim 1. *Let s be a schedule in which $s_j = M_i$. It is beneficial for job j to migrate from machine M_i to machine $M_{i'}$ if and only if $L_{i'}(s) + p_j > L_i(s)$ and $B > \gamma$ (load-increasing migration) or $L_{i'}(s) + p_j < L_i(s)$ and $B < \gamma$ (load-decreasing migration), where $\gamma = (L_i(s)(L_{i'}(s) + p_j))/p_j$.*

The following observations provide lower and upper bounds for a job's individual cost.

Observation 1. In any joint action s, for every job j, $c_j(s) \geqslant 2\sqrt{p_j B}$. Additionally, for every j such that $p_j \geqslant B$, $c_j(s) \geqslant p_j + B$.

Proof. The cost of j when assigned together with a (possible empty) set of jobs with total load l is $c_j(s) = p_j + l + p_j B/(p_j + l)$. Simple calculus shows that (i) if $p_j \leqslant B$, then this term gets its minimal value for $l = \sqrt{p_j B} - p_j$, for which $c_j(s) = 2\sqrt{p_j B}$; and (ii) if $p_j \geqslant B$, then $c_j(s)$ is minimized for $l = 0$, for which $c_j(s) = p_j + B$. □

Because a job of length p_j can always migrate to a dedicated machine and incur a cost $p_j + B$, the following observation follows trivially:

Observation 2. In any NE s, for every job j, $c_j(s) \leqslant p_j + B$.

The following observation, which can be easily derived from Claim 1, provides some insight into beneficial and nonbeneficial job migrations.

Observation 3. A job j of length $p_j < B$ that is assigned to a machine with load smaller than B cannot reduce its cost by migrating to a machine with load greater than B or to a dedicated machine.

Finally, one can easily verify that if the total load of jobs does not exceed B, assigning all jobs into a single machine is an NE.

Observation 4. If $B \geqslant \sum_j p_j$, then the schedule s in which all jobs are scheduled on a single machine is an NE.

2.3. Longest Processing Time (LPT) Rule

LPT is a well-known scheduling heuristic due to Graham (1969). The LPT rule sorts the jobs in a nonincreasing order of their lengths and greedily assigns each job to the least-loaded machine. In the traditional load-balancing problem, where the number of machines is fixed and the incurred cost includes only the load on the machine, the LPT rule is known to produce an NE (Fotakis et al. 2002). However, if the number of machines is not limited by a fixed number, the standard LPT will simply assign each job to a dedicated machine, which will not necessarily yield an NE in the presence of activation costs.

A natural generalization of LPT that assigns each job to a machine that minimizes its cost (including both the load and the activation cost components) does not necessarily lead to an NE either, even with unit-length jobs. Consider, for example, a game with four unit-size jobs and with activation cost $B = 4 - \varepsilon$. A greedy assignment, trying to minimize the cost of a newly assigned job will result in a schedule on two machines with respective loads 3 and 1, which is not an NE.

In this paper we show that employing LPT with the "right" number of machines would produce an NE. Establishing the above assertion requires the following nontrivial property of the LPT algorithm. Recall that $L_{\max}(s) = \max_i L_i(s)$ denotes the makespan of a schedule s.

Lemma 1. *Let I be a set of n jobs of sizes $p_1, \ldots, p_n$, and let C be some positive real number satisfying $p_j < C$ for every j and $C < \sum_j p_j$. Let s_k be a schedule of I obtained from LPT with k machines, and denote by m the minimal number of machines such that $L_{\max}(s_m) \leqslant C < L_{\max}(s_{m-1})$. Then $L_i(s_m) + L_{i'}(s_m) > C$ for every $i \neq i'$, $i, i' \in [m]$.*

Proof. First, note that because $C < \sum_j p_j$ it must be that $m > 1$. For $m = 2$ the lemma trivially holds: if LPT fails to assign the jobs on a single machine having makespan at most C, then the total load of the jobs must be more than C. Therefore, in the sequel we assume that $m > 2$. Assume by way of contradiction that the assertion of the lemma is false and let I be a minimal (in terms of the number of jobs) set of jobs contradicting the lemma. That is, there are two machines in the schedule s_m whose total load is at most C. Let $p_{\min}$ denote the length of the shortest job in I.

Claim 2. *The most-loaded machine in s_m has at least two jobs.*

Proof. Assume by way of contradiction that the most-loaded machine in s_m has a single job of length p_j. It is easy to see that $p_j \geqslant p_{j'}$ for every $j' \in I$.

Consider the schedule s_{m-1}. We claim that job j must be assigned in s_{m-1} with at least one additional job. To see this, consider the set $I' = I \backslash \{j\}$. We show that if job j is assigned in s_{m-1} with no additional jobs, then I' also contradicts the lemma—contradicting the minimality of the instance I. Given that j is assigned alone in s_m, the assignment of I' by LPT on $m-1$ machines is identical to its assignment as part of I in s_m, and therefore has makespan at most C. Similarly, if j is assigned alone in s_{m-1}, then the assignment of I' on $m-2$ machines is identical to its assignment as part of I in s_{m-1} and, therefore, it has makespan larger than C. By the contradiction assumption, the two most lightly loaded machines in s_m have total load at most C. Given that $m > 2$, it is feasible to apply LPT on I' and $m-2$ machines. Moreover, the machine holding job j, which is the most-loaded machine in s_m, is not one of the two lightly loaded machines in s_m. Therefore, these two machines have exactly the same load as in the schedule of I' on $m-1$ machines. Therefore, I' contradicts the lemma, contradicting the minimality of the instance I. We conclude that job j is not assigned alone in s_{m-1}.

Given that a second job is assigned to j's machine in s_{m-1}, it must be that all other $m-2$ machines have load at least p_j (else, LPT would not add the second job to j'-s machine). Therefore, in s_{m-1}, the load on every machine is at least p_j. We also know that there exists a machine of load greater than C. We conclude that sum of the jobs' sizes, denoted P, must satisfy $P > C + (m-2)p_j$.

Now consider s_m again. By the contradiction assumption, there exist two machines with total load of at most C and the makespan is p_j. It follows that $P \leqslant C + (m-2)p_j$, in contradiction to the above. □

We next establish a lower bound on the makespan of s_m. Because $L_{\max}(s_{m-1}) > C$ and all the jobs are shorter than C, the most-loaded machine in s_{m-1} holds at least two jobs. Moreover, the most-loaded machine in s_{m-1} holds the shortest job, of length $p_{\min}$; otherwise, by removing from I the jobs shorter than $p_{\min}$ we get a smaller instance contradicting the minimality of I. By LPT, it holds that for every machine i, $L_i(s_{m-1}) \geqslant L_{\max}(s_{m-1}) - p_{\min} > C - p_{\min}$. In addition, there exists a machine with load greater than C. Therefore, the sum of the jobs' sizes, P, is greater than $(m-2)(C - p_{\min}) + C = (m-1)C - (m-2)p_{\min}$.

On the other hand, by the contradiction assumption, in s_m there are two machines with total load of at most C. Therefore, the total load on the remaining $(m-2)$ machines is at least $P - C > (m-2)(C - p_{\min})$. Thus, there must exist a machine with load greater than $C - p_{\min}$. We obtain the following lower bound:

$$L_{\max}(s_m) > C - p_{\min}. \tag{1}$$

We next provide a lower bound on the load of any machine in s_m. Let p_m denote the length of the mth-longest job in I, and let l be the last job on the most-loaded machine in s_m. By Claim 2 there exists a job at least as long as job l on this machine. Because LPT first assigns a single job on each machine, $p_l \leqslant p_m$. Moreover, by LPT, the load on every machine in s_m is at least $L_{\max}(s_m) - p_l$. We obtain the following inequality for every machine i:

$$\begin{aligned} L_i(s_m) &\geqslant L_{\max}(s_m) - p_l \\ &\geqslant L_{\max}(s_m) - p_m > C - p_{\min} - p_m, \end{aligned} \tag{2}$$

where the last inequality follows from Equation (1).

In s_m, the first job on every machine has a load of at least p_m, whereas every other job has a load of at least $p_{\min}$. Thus, the load of every machine in s_m with at least two jobs is at least $p_m + p_{\min}$. By the assumption that there exist two machines in s_m with total load of at most C, the least-loaded machine has a load of at most $C/2$. By Equation (2), its load is greater than $C - p_{\min} - p_m$. This implies that:

$$p_l + p_{\min} > C/2. \tag{3}$$

Let M_1, M_2 denote the least-loaded and the second-least-loaded machine in s_m, respectively. Distinguish between three cases:

Case 1: M_1 holds at least two jobs. Recall that the load of every machine with at least two jobs is at least $p_m + p_{\min}$, which is greater than $C/2$ by Equation (3). Therefore, the total load on any two machines exceeds C.

Case 2: M_1 holds a single job and M_2 holds at least two jobs. In this case, the load of M_2 is at least $p_m + p_{\min}$, and the load on M_1 is greater than $C - p_{\min} - p_m$ (by Equation (2)). Summed together, their total load exceeds C.

Case 3: M_1 and M_2 each hold a single job. We distinguish between two cases:

1. $|I| \leqslant 2(m-1)$: A known property of LPT is that if the total number of jobs is at most twice the number of machines, then LPT produces a schedule of minimum makespan (Graham 1969). Specifically, if $|I| \leqslant 2(m-1)$, then any schedule of I on $m-1$ machines has makespan of at least $L_{\max}(s_{m-1}) > C$. Consider the schedule s'_{m-1}, induced by s_m by merging the jobs on M_1 and M_2 into a single machine. It holds that $L_{\max}(s'_{m-1}) = \max\{L_1(s'_{m-1}) + L_2(s'_{m-1}), L_{\max}(s_m)\}$. Because $L_{\max}(s_m) \leqslant C$, it must hold that $L_1(s'_{m-1}) + L_2(s'_{m-1}) > C$.
2. $|I| > 2(m-1)$: Because M_1, M_2 each hold a single job, there are $|I| - 2 > 2(m-2)$ jobs that are assigned to the other $m-2$ machines, implying that there must exist a machine holding at least three jobs. The first of these jobs is of size at least p_m, and the second is of size at least $p_{\min}$. Given that LPT assigned a third job to this machine and not to M_1, it must be that the load on M_1 at the time of the assignment was at least $p_m + p_{\min}$, which is greater than $C/2$ by Equation (3).

We conclude that the total load on any two machines is greater than C, in contradiction to the assumption. This establishes the assertion of the lemma. □

3. Equilibrium Existence and Computation

Under the uniform sharing rule, a pure NE might not exist. Consider, for example, an instance with two jobs of sizes 1, 10, and with activation cost $B = 4$. On dedicated machines, the jobs' costs are 5 and 14, respectively. If they are assigned together, they both pay 13. Thus, no schedule is stable: the short job will escape to a dedicated machine, whereas the long job will always join it. In contrast, under the proportional sharing rule a pure NE always exists. In what follows, we present an algorithm that computes an NE in polynomial time.

Algorithm LPT*:

1. Assign each job j such that $p_j \geqslant B$ (henceforth *long* jobs) to a dedicated machine.
2. Assign the remaining jobs (henceforth *short* jobs) by algorithm LPT to m machines, where m is the minimal number of machines such that LPT produces a schedule with makespan at most B.

Observe that the number of machines used in the second step is well defined: because all the participating jobs are shorter than B, a schedule having makespan less than B exists. The running time of LPT* is $O(n \log^2 n)$. In particular, long jobs are identified and scheduled in time $O(n)$, the short jobs are sorted in time $O(n \log n)$, and then LPT is executed at most $\log n$ times (binary search for the right value of m—which is an integer in the range $[1, n]$). We next prove that Algorithm LPT* produces an NE.

Theorem 5. *Every profile obtained by algorithm LPT* is an NE.*

Proof. Let s be a profile that is obtained by algorithm LPT*. We show that no unilateral migration from s is beneficial. By Observation 1, no long job can benefit from migration; and by Observation 3, no short job can benefit from joining a long job or from activating a new machine. It remains to show that no short job can benefit from migrating to another LPT machine. If the total load of short jobs is at most B, then LPT* assigns them to a single machine; and no migrations among LPT machines are possible.

If the total load of short jobs is more than B, then LPT* assigns them to multiple machines. We show that no migrations among these machines is beneficial. Let j be a short job assigned to M_i. We show that it cannot benefit by migrating to some LPT machine $M_{i'}$. A known property of LPT is that the load of $M_{i'}$ after the migration is at least the load of M_i in s; i.e., $L_{i'}(s) + p_j \geqslant L_i(s)$. Consequently, only load-increasing migrations should be considered. By Claim 1, it suffices to show that $B \leqslant (L_i(s)(L_{i'}(s)+p_j))/p_j = (L_i(s)L_{i'}(s))/p_j + L_i(s)$. Because $L_i(s) \geqslant p_j$, it follows that $(L_i(s)L_{i'}(s))/p_j + L_i(s) \geqslant L_{i'}(s) + L_i(s) \geqslant B$, where the last inequality follows from Lemma 1. □

Remark 1. We defer to the appendix a simpler proof for the existence of a pure NE under the proportional sharing rule. That proof shows that any lexicographically minimal assignment to m machines such that m is the minimal number of machines for which the lexicographically minimal assignment does not exceed B is an NE. Although that other proof is simpler, the current algorithm is superior in two ways. First, finding a lexicographically minimal assignment is an NP-hard problem. Its hardness follows from the fact that every lexicographically minimal assignment minimizes the makespan, and makespan minimization is known to be NP-hard (Garey and Johnson 1979). In contrast, as stated above, algorithm LPT* computes an NE in polynomial time. Second, the current result establishes the new property of the (standard) LPT algorithm, presented in Lemma 1, which may be of interest independently of its game-theoretic interpretation.

Remark 2. An instance might have several different Nash equilibria. For example, for any sufficiently large integer k, consider an instance with $k!$ unit-length jobs and with activation cost $B = k!$. For every number of machines $m \in \{1, \ldots, k\}$, a balanced assignment of $k!/m$ jobs to each of the machines is an NE. All of the jobs experience the same cost of $k!/m + m$. Migrating to a new machine would result in cost $k! + 1 \geqslant k!/m + m$ and is therefore nonbeneficial. Migrating to another open machine would result in cost $k!/m + 1 + k!/(k!/m + 1) > k!/m + m$ and is also nonbeneficial. Thus, this game has at least k different NE (corresponding to the different number of machines that are activated).

Remark 3. Although an NE can be found in polynomial time, finding a best Nash equilibrium is an NP-hard problem, as we show in §5.1.

4. Convergence of Best-Response Dynamics

Best-response dynamics (BRD) is a local search method where in each step some player is chosen and plays its best-response strategy, given the strategies of the others. In systems where agents repeatedly perform improvement steps until they reach a Nash equilibrium, the notion of a pure NE is well justified. In order to better understand the behavior of these dynamics, this section explores the convergence rate of best-response dynamics into a pure NE.

We first show that unlike other job-scheduling games, in our model BRD do not necessarily converge to a Nash equilibrium.

Observation 6. Under the proportional sharing rule BRD might not converge to an NE.

Proof. Consider the instance with four jobs of lengths 10, 10, 10, 20, and $B = 72$. In a machine with two jobs of length 10, each of the jobs incurs a cost of $36 + 20 = 56$. In a machine with jobs of lengths 10 and 20, the jobs of lengths 10 and 20 cost $24+30=54$ and $48+30=78$, respectively. In a machine with jobs of lengths 10, 10, 20, each of the jobs of length 10 incurs a cost of $18+40=58$, whereas the long job's cost is $36+40=76$. Consider a profile in which two jobs of length 10 are assigned to one machine and jobs of lengths 10, 20 are assigned to a second machine. The long job can reduce its cost by migrating to the other machine. Then, one of the jobs of length 10 assigned to the machine that the long job joined can benefit by migrating to the other machine. That brings the system back to the initial configuration. Thus, BRD are not guaranteed to converge. $\square$

Note that in the last example, BRD will not converge to an NE under the given initial state, independent of the order in which the agents respond.

In contrast to the case of heterogeneous jobs, if all the jobs have the same length, then the induced game is equivalent to a congestion game in which the maximal number of resources is n (Rosenthal 1973). Without loss of generality, we assume that $p_j = 1$, i.e., the case of unit-length jobs; the activation cost can be scaled accordingly. In the case of unit-size jobs, one can easily verify that the function $\Phi(s)=\sum_i (B\cdot H_{l_i}+\frac{1}{2}l_i^2)$, where l_i denotes the number of jobs on machine i, $H_0=0$, and $H_k=1+1/2+\cdots+1/k$, is a potential function for the game. Convergence to an NE is guaranteed in potential games, but the convergence time might be exponential.

In many cases, the convergence time depends on the order in which the players apply their best-response moves. In a follow-up work (Feldman and Tamir 2012), we study the convergence time of BRD in the setting introduced in this work for the case of unit-length jobs. We show that the convergence of BRD might take $\Omega(n\log(n/B))$ moves, but we also introduce a specific form of BRD that converges faster. In particular, we show that if at every time step, a job that incurs the highest cost is chosen to perform its best response, then convergence occurs within a linear number of moves.

5. Equilibrium Quality

In this section we study the inefficiency caused due to strategic behavior, as quantified by the price of anarchy (PoA) and price of stability (PoS) measures. We compute the PoA and PoS with respect to the objective of minimizing the highest cost among all the jobs; that is, given a profile s, the social cost of s is given by $g(s)=\max_j c_j(s)$. All of the results in this section refer to the proportional sharing rule.

We provide tight parametric bounds on the PoA and PoS, which depend on the ratio between the activation cost (B) and the longest job. Formally, let αB be the length of the longest job in the instance.

We first observe that if there exists a job of length at least B, then every NE is optimal.

Observation 7. If there exists a job j such that $p_j \geqslant B$ (i.e., $\alpha \geqslant 1$), then $PoA=1$.

Proof. Let $j'=\arg\max_j p_j$. In particular, $p_{j'}\geqslant B$. By Observation 2, for every NE profile s and every job j, $c_j(s)\leqslant p_j+B$. Therefore, $\max_j c_j(s)\leqslant p_j+B\leqslant p_{j'}+B$. On the other hand, in any schedule, and in particular the optimal profile s^*, $\max_j c_j(s^*)\geqslant c_{j'}(s^*)\geqslant p_{j'}+B$, where the last inequality follows from Observation 1. Thus, $PoA=1$ as required. $\square$

In light of the last observation, the interesting case is where $\alpha<1$, which is the focus of the remainder of this section. The following theorem provides a tight bound on the PoA as a function of α.

Theorem 8. *For every $\alpha<1$, it holds that $PoA\leqslant (1+\alpha)/(2\sqrt{\alpha})$. In addition, for every $\alpha<1$, there exists an instance such that $PoA=(1+\alpha)/(2\sqrt{\alpha})$.*

Proof. We first provide an upper bound for the PoA. By Observation 1, the cost of a job j is at least $2\sqrt{p_jB}$; in particular, the cost of the longest job is at least $2\sqrt{\alpha}B$. On the other hand, by Observation 2, $c_j(s)\leqslant p_j+B$ for every job j and NE profile s. Therefore, $PoA\leqslant (B+\alpha B)/(2\sqrt{\alpha}B)=(1+\alpha)/(2\sqrt{\alpha})$.

For the lower bound, consider an instance with B unit-length jobs. That is, $\alpha=1/B$. An optimal schedule assigns the jobs in groups of $\sqrt{B}$ jobs to each machine (w.l.o.g., B is a square of some integer). In this schedule, the cost of every job is $2\sqrt{B}$. However, a schedule that assigns all the jobs to a single machine is also an NE, because each job incurs a cost of $B+1$, which cannot be reduced by migrating to a new machine. We get: $PoA\geqslant (B+1)/(2\sqrt{B})=(1+\alpha)/(2\sqrt{\alpha})$. $\square$

The term in the last theorem (which bounds the PoA) tends to infinity as α gets closer to zero. Thus, there is no universal bound on the price of anarchy. We note that the bounds for the PoA are identical for the objective of minimax and the minisum objective (studied in Chen and Gürel 2012). This is explained by the fact that the worst NE is achieved with one machine and identical jobs, in which case all jobs have the same cost, and therefore the average and the maximal cost are the same.

In contrast, the following couple of theorems show that the price of stability is always smaller than 5/4, and this is tight.

Theorem 9. *For every instance of the problem, it holds that $PoS<5/4$.*

Proof. Let $p=\alpha B$ be the length of the longest job in the instance, for $\alpha<1$. If $\alpha>1/4$, then we show that the *Price of Anarchy* is less than 5/4. By Observation 1, the cost of the longest job is at least $2\sqrt{pB}$; thus, $OPT\geqslant 2\sqrt{pB}$. On the other hand, by Observation 2, $c_j(s)\leqslant p_j+B\leqslant p+B$ for every job j and NE profile s. It follows that $PoA\leqslant (B+p)/(2\sqrt{pB})=(1+\alpha)/(2\sqrt{\alpha})$.

It is easy to verify that the last expression is smaller than 5/4 for $\alpha > 1/4$. The assertion follows because $PoS \leqslant PoA$. Note that in this case an NE achieving $PoS < 5/4$ can be computed by Algorithm LPT*.

Otherwise, $\alpha \leqslant 1/4$. Let m be the minimal number of machines such that algorithm LPT on m machines produces a schedule whose makespan is at most $2(\sqrt{\alpha}-\alpha^2)B$, and let s be a profile obtained by LPT on m machines. We distinguish between two cases:

Case 1: The total processing time of the jobs is at most $2(\sqrt{\alpha}-\alpha^2)B$. In this case, all the jobs are assigned to a single machine, i.e., $m=1$, and this is an NE by Observation 4. The maximal cost is realized by the longest job (of length αB). Let $f(P)=P+(\alpha B^2)/P$ denote the cost of the longest job as a function of the total processing time P. If $P<\sqrt{\alpha}B$, then $f(P)$ decreases in P, and assigning all the jobs to a single machine achieves the optimal cost; thus, $PoS=1$. If $P \geqslant \sqrt{\alpha}B$, then $f(P)$ increases in P, and it suffices to bound the maximal cost for $P=2(\sqrt{\alpha}-\alpha^2)B$. It follows that

$$PoS \leqslant \frac{f(2(\sqrt{\alpha}-\alpha^2)B)}{OPT} \leqslant \frac{f(2(\sqrt{\alpha}-\alpha^2)B)}{2\sqrt{\alpha}B} < \frac{5}{4},$$

where the inequality $OPT \geqslant 2\sqrt{\alpha}B$ follows from Observation 1, and the last inequality holds for every $\alpha \leqslant 1/4$.

Case 2: The total processing time of the jobs is more than $2(\sqrt{\alpha}-\alpha^2)B$. In this case, $m \geqslant 2$. We first provide a lower bound for the load of any machine.

Claim 3. *The load on any machine in the profile s is greater than $(\sqrt{\alpha}-\alpha/2-\alpha^2)B$.*

Proof. By Lemma 1 (applied with $C=2(\sqrt{\alpha}-\alpha^2)B$), the total load on any two machines must be greater than $2(\sqrt{\alpha}-\alpha^2)B$. Assume by way of contradiction that the load on some machine is at most $(\sqrt{\alpha}-\alpha/2-\alpha^2)B$. Because s is produced by LPT, the gap in the load between any two machines cannot exceed the length of the longest job, which is αB. Therefore, the load of any other machine is at most $(\sqrt{\alpha}-\alpha/2-\alpha^2)B+\alpha B$. We get that the total load on these two machines is at most $2(\sqrt{\alpha}-\alpha^2)B$, in contradiction to Lemma 1. □

Next, we show that the profile s is an NE.

Claim 4. *The profile s is an NE.*

Proof. Observe that for any $\alpha \leqslant 1/4$, $2(\sqrt{\alpha}-\alpha^2)<1$; thus, the makespan is less than B. Therefore, by Observation 3, no job will migrate to a dedicated machine. Because in schedules obtained by LPT unilateral migrations cannot decrease the load, and because a job is indifferent if its load does not change, it suffices to show that no load-increasing deviation is profitable. It also follows from LPT that the gap between any two machines cannot exceed the length of the longest job, i.e., αB, and by Lemma 1 (applied with $C=2(\sqrt{\alpha}-\alpha^2)B$), the total load on any two machines is greater than $2(\sqrt{\alpha}-\alpha^2)B$. Given a lower bound on the sum of the two loads, and an upper bound on the difference between the two loads, one can obtain the following bound on their multiplication. Specifically, for any two machines M_i, $M_{i'}$,

$$\begin{aligned} L_i(s)L_{i'}(s) &\geqslant \left(\sqrt{\alpha}-\alpha^2-\frac{\alpha}{2}\right)B\left(\sqrt{\alpha}-\alpha^2+\frac{\alpha}{2}\right)B \\ &= \left(\alpha-2\sqrt{\alpha}\alpha^2+\alpha^4-\frac{\alpha^2}{4}\right)B^2. \end{aligned} \tag{4}$$

By Claim 1 a load-increasing migration from load $L_i(s)$ to load $L_{i'}(s)$ is profitable for a job of length p_j only if $B > L_i(s)(L_{i'}(s)+p_j)/p_j$. However,

$$\begin{aligned} &\frac{L_i(s)(L_{i'}(s)+p_j)}{p_j} \\ &\quad = \frac{L_i(s)L_{i'}(S)}{p_j}+L_i(s) \\ &\quad > \left(1-2\sqrt{\alpha}\alpha+\alpha^3-\frac{\alpha}{4}\right)B+\left(\sqrt{\alpha}-\frac{\alpha}{2}-\alpha^2\right)B \\ &\quad = \left(1-\sqrt{\alpha}(2\alpha-1)-\frac{3\alpha}{4}+\alpha^3-\alpha^2\right)B, \end{aligned}$$

where the inequality follows from Equation (4) and Claim 3. The last expression is greater than B for every $\alpha \leqslant 1/4$. □

Let M_i be the machine on which the maximal cost is achieved. Because $p_j \leqslant \alpha B$ for every job j,

$$\max_j c_j(s) = \max_{j:s_j=M_i} c_j(s) \leqslant L_i(s)+\frac{\alpha B^2}{L_i(s)}.$$

Let $f(x)=x+(\alpha B^2)/x$. The function $f(x)$ decreases in x for every $x<\sqrt{\alpha}B$ and increases in x for every $x>\sqrt{\alpha}B$. Because for every i, $L_i(s)\leqslant 2(\sqrt{\alpha}-\alpha^2)B$ (by the construction of s) and $L_i(s)>(\sqrt{\alpha}-\alpha/2-\alpha^2)B$ (by Claim 3), it suffices to bound the maximal cost at these two points. Because $OPT \geqslant 2\sqrt{\alpha}B$, it follows that

$$PoS \leqslant \max\left(\frac{f(2(\sqrt{\alpha}-\alpha^2)B)}{2\sqrt{\alpha}B}, \frac{f((\sqrt{\alpha}-\alpha/2-\alpha^2)B)}{2\sqrt{\alpha}B}\right).$$

The assertion of the theorem follows by observing that for every $\alpha \leqslant 1/4$ it holds that $(f(2(\sqrt{\alpha}-\alpha^2)B))/(2\sqrt{\alpha}B) < 5/4$, and $(f((\sqrt{\alpha}-\alpha/2-\alpha^2)B))/(2\sqrt{\alpha}B) \leqslant 1.1125 < 5/4$. □

Remark 1. The last proof, in addition to bounding the PoS, also provides a polynomial-time algorithm for finding an NE with cost less than factor 5/4 from the social optimum.

Remark 2. The last result should be contrasted with load-balancing games in which a job's cost equals its machine's load, where there always exists a social optimum that is an NE (Vöcking 2007), and thus the price of stability is 1.

We next show that the above upper bound is tight. That is, for every $\varepsilon > 0$, there exists an instance for which the ratio between the best NE and the social optimum is greater than $5/4 - \varepsilon$. Interestingly, this ratio can be achieved with unit-length jobs.

Theorem 10. *For every $\varepsilon > 0$ there exists a job-scheduling game with unit-length jobs for which $PoS > 5/4 - \varepsilon$.*

Proof. Let b be an integer larger than $2/\varepsilon$, and let $B = b^2$. Consider an instance with $n = 2b - 2$ unit-length jobs. We claim that the game induced by this instance has a unique NE in which all the jobs are assigned to a single machine. Because $n < B$, it follows from Observation 4 that this is an NE. To prove uniqueness, assume by way of contradiction that there is an NE schedule s in which the number of activated machines is at least 2, and let n_i denote the number of jobs assigned to machine M_i in s. Having unit-length jobs, the cost of a job that is assigned to a machine with load k is $c(k) = k + B/k$. If $m = 2$, then for some $r > 0$, $n_1 = b - r$ and $n_2 = n - n_1 = b - 2 + r$. It is easy to verify that for any $r > 0$, $c(b - r) > c(b - 1 + r)$; thus, jobs on the first machine benefit from migrating to the second machine, contradicting the assumption that this is an NE. If $m > 2$, then because $n = 2b - 2$, there must exist at least two machines with at most $b - 1$ jobs; denote them i, i'. It is easy to verify that for any $0 < n_i \leqslant n_{i'} \leqslant b - 1$ it holds that $c(n_i) > c(n_{i'} + 1)$. Therefore, a unilateral migration from machine M_i to machine $M_{i'}$ is beneficial, in contradiction to s being a Nash equilibrium. It follows that assigning all the jobs to a single machine is the unique NE. The cost of each job in this schedule is $c(2b - 2) = 2b - 2 + B/(2b - 2)$.

Consider next a schedule of this instance with two activated machines, each holding $b - 1$ jobs. The cost of each job in this schedule is $c(b - 1) = b - 1 + B/(b - 1)$. It follows that:

$$PoS \geqslant \frac{2b - 2 + B/(2b - 2)}{b - 1 + B/(b - 1)} = \frac{5b^2 - 8b + 4}{4b^2 - 4b + 2}.$$

One can easily verify that the last expression is greater than $5/4 - \varepsilon$ for every $b > 2/\varepsilon$. This establishes the assertion of the theorem. Note that the ratio approaches 5/4 as the ratio between the job's length and the activation cost increases. □

Remark. Although the focus here is on pure Nash equilibria, our results have direct implications on the price of anarchy and the price of stability with respect to mixed Nash equilibria. In the general case, because the PoA is unbounded already with respect to pure strategies, then it is trivially unbounded with respect to mixed strategies as well. In addition, for the special case in which there exists a job whose length exceeds (or equals) B, it is not too difficult to extend Observation 7, which shows that the PoA is 1, to work with respect to mixed strategies as well. Regarding the price of stability, our positive results (i.e., a tight bound of 5/4) trivially carry over to mixed NE (because the best mixed NE is at least as good as the best pure NE, by definition). Finding a tight bound for the price of stability under mixed equilibria is left as an open problem.

5.1. Computational Hardness of the Best Nash Equilibrium

Although an NE can be found in polynomial time, finding a best Nash equilibrium is an NP-hard problem, as derived from the following theorem.

Theorem 11. *Given an instance of our game and some $c \in \mathbb{R}$, it is NP-hard to determine whether there exists an NE that achieves social cost less than or equal to c.*

Proof. We show a reduction from the *Partition* problem. The input of *Partition* is a set A of n integers $a_1, \ldots, a_n$ with a total size of $2S$ for some integer S. The goal is to decide whether A contains a subset of total size S. Given A, consider the game instance consisting of n jobs having lengths $a_1, \ldots, a_n$, and machines with activation cost $B = S^2/a_{\max}$, where $a_{\max}$ is the length of the longest job, and let $c = 2S$.

We show that the game admits an NE with a social cost of at most $2S$ if and only if A admits a partition. Assume first that A admits a partition, and consider a schedule s on two machines, each having a load of S. Because all the jobs incur the same load, the maximal cost is incurred by the longest job and is given by $S + a_{\max}B/S = 2S$. It remains to show that s is an NE. Consider a job j of length a_j. By Claim 1, moving to an empty machine is beneficial only if $B < S$; that is, if $S < a_{\max}$, which is false by the existence of a partition. Moving to the nonempty machine is beneficial, by Claim 1, only if $B > S(S + a_j)/a_j$; that is, if $S/a_{\max} > (S + a_j)/a_j$, which is clearly false. It follows that s is an NE.

In the other direction, assume that A does not admit a partition. This implies that in any NE, the longest job is assigned to a machine with load $l \neq S$. The cost of the longest job is given by $l + a_{\max}B/l$, which achieves its minimal value for $l = \sqrt{a_{\max}B} = S$. Therefore, for any $l \neq S$, the cost of the longest job is strictly greater than c. The assertion of the theorem is established. □

Remark. Note that the proof of Theorem 9 essentially suggests a polynomial-time algorithm for finding an NE that approximates the social optimum within a factor of 5/4. Clearly, such an NE also approximates the best NE within a factor of 5/4. From the lower bound established in Theorem 10, it follows that this is the best approximation to the social optimum that can be found. The problem of approximating the best NE within a factor smaller than 5/4 remains open. We also note that the NE schedule computed by Algorithm LPT*, given in §3, as well as the NE schedule produced by a lexicographically minimal assignment, as described in the appendix, do not approximate the social welfare well. In particular, for the instance that is used to prove the lower bound of the PoA (given in the proof of Theorem 8), both LPT* and the lexicographically minimal assignment result in the worst NE.

6. Conclusions and Open Problems

The current work considers job-scheduling settings with conflicting congestion effects, and establishes results with respect to NE existence, computation, quality, and convergence. Our results indicate that the new setting of conflicting congestion effects significantly affects the properties of the obtained game. Our analysis and results suggest several intriguing open questions for future work.

1. It is desirable to generalize our results to other job-scheduling settings with different machines models, different sharing rules, different cost structures, and different social choice functions.

2. It would be interesting to provide a characterization of instances or initial states from which best-response dynamics (or other types of dynamics, such as better-response dynamics) are guaranteed to converge to a Nash equilibrium.

3. It is desirable to study this model with respect to additional notions of equilibria. In particular, studying the existence and quality of a strong equilibrium (where stability is guaranteed against coalitional deviations) would be interesting.

4. We proved that finding the best Nash equilibrium is an NP-hard problem. The proof of Theorem 9 essentially suggests a polynomial-time algorithm for finding an NE that approximates the social optimum, and therefore also the best NE within a factor of 5/4. The problem of approximating the best NE within a factor smaller than 5/4 remains open.

Appendix. Equilibrium Existence

In this section we prove that any lexicographically minimal assignment of the short jobs on m machines, such that m is the minimal number of machines for which the lexicographically minimal assignment of these jobs does not exceed B, is an NE. We first define a complete order on the profiles.

Definition 2. A vector $(L_1, L_2, \ldots L_m)$ is smaller than $(\hat{L}_1, \hat{L}_2, \ldots \hat{L}_m)$ *lexicographically* if for some i, $L_i < \hat{L}_i$ and $L_k = \hat{L}_k$ for all $k < i$. A profile s is smaller than s' lexicographically if the vector of machine loads $L(s) = (L_1(s), \ldots, L_m(s))$, sorted in nonincreasing order, is smaller lexicographically than $L(s')$, sorted in nonincreasing order.

We use the following property of the lexicographically minimal assignment.

Claim 5. *Let s be a lexicographically minimal assignment, and suppose that job j is assigned to machine M_i. Then, $L_{i'}(s) + p_j \geqslant L_i(s)$ for every $i' \neq i$.*

Proof. If $L_{i'}(s) \geqslant L_i(s)$, the statement holds trivially. Otherwise (i.e., $L_{i'}(s) < L_i(s)$), suppose by way of contradiction that $L_{i'}(s) + p_j < L_i(s)$; then, the schedule that assigns the job of length p_j to the machine of load $L_{i'}(s)$ produces a lexicographically smaller assignment, in contradiction to the minimality of s. □

Given an instance of jobs I, let $I_{short} \subseteq I$ be the subset of jobs having length less than B. Let $\hat{s}_k$ be the lexicographically minimal assignment of I_{short} on k machines. Let m be such that the makespan under $\hat{s}_m$ is smaller than B, whereas the makespan under $\hat{s}_{m-1}$ is at least B. Note that m is well defined, because all the participating jobs are shorter than B.

Let $\hat{s}$ be the profile in which (i) every j s.t. $p_j \geqslant B$ is assigned to a dedicated machine and (ii) the jobs of I_{short} are assigned according to $\hat{s}_m$.

Theorem 12. *The profile $\hat{s}$ is an NE.*

Proof. We show that none of the possible migrations is beneficial. By Observation 1, no long job can benefit from migration, and by Observation 3 no short job can benefit from joining a long job or from activating a new machine. It remains to show that no short job can benefit from migrating to another machine of $\hat{s}_m$. Let j be a short job assigned to M_i. By Claim 5, $L_{i'}(\hat{s}) + p_j \geqslant L_i(\hat{s})$ for every $i' \neq i$. In other words, only load-increasing migration can be performed. By Claim 1, such migrations are beneficial for job j if and only if $B > (L_i(\hat{s})(L_{i'}(\hat{s}) + p_j))/p_j$. Assume $p_j = \alpha L_i(\hat{s})$ for some $0 < \alpha \leqslant 1$. Then the migration condition can be written as $B > (L_i(\hat{s})(L_{i'}(\hat{s}) + \alpha L_i(\hat{s})))/(\alpha L_i(\hat{s})) = (L_{i'}(\hat{s}))/\alpha + L_i(\hat{s}) \geqslant L_{i'}(\hat{s}) + L_i(\hat{s})$. Therefore, to obtain a contradiction, it is sufficient to show that for every i, i', $L_i(\hat{s}) + L_{i'}(\hat{s}) \geqslant B$. Assume by way of contradiction that there exist i, i', s.t. $L_i(\hat{s}) + L_{i'}(\hat{s}) < B$. Consider the schedule that is identical to $\hat{s}$ except the jobs assigned to machines i and i' are jointly assigned to a single machine. This schedule produces a makespan smaller than B on $m-1$ machines, which is a contradiction to the choice of m. □

Acknowledgments

An extended abstract of this paper appeared in the *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE)*, December 2008. This work was partially supported by the Israel Science Foundation [Grant 1219/09], by the Leon Recanati Fund of the Jerusalem School of Business Administration, the Google Inter-university Center for Electronic Markets and Auctions, and the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA [Grant 274919].

Endnotes

1. Jobs, players, and users are used interchangeably throughout the paper.
2. The concept of activation cost has been studied in a cooperative game-theoretic setting of *facility location* by Jain and Mahdian (2007). In the facility location game, each facility has an activation cost, and the cost of each set of players is the sum of the activation cost of the facilities and the cost of connecting every player to an opened facility. However, facility location games have been studied mainly within the framework of cooperative game theory, which is very different from ours. Also, the individual cost of a player is independent of the players sharing a facility with him.

References

Albers S, Elits S, Even-Dar E, Mansour Y, Roditty L (2006) On Nash equilibria for a network creation game. *Annual ACM-SIAM Sympos. Discrete Algorithms* (*SODA*) (ACM, New York).

Andelman N, Mansour Y, Feldman M (2009) Strong price of anarchy. *Games Econom. Behav.* 65(2):289–317.

Anshelevich E, Dasgupta A, Kleinberg JM, Tardos É, Wexler T, Roughgarden T (2004) The price of stability for network design with fair cost allocation. *Sympos. Foundations Comput. Sci.* (*FOCS*) (IEEE Computer Society), 295–304,

Awerbuch B, Azar Y, Epstein A (2005) The price of routing unsplittable flow. 37*th ACM Sympos. Theory Comput.* (ACM, New York).

Bernstein F, Federgruen A (2001) Decentralized supply chains with competing retailers under demand uncertainty. *Management Sci.* 51(1):18–29.

Bonifaci V, Harks T, Schafer G (2010) Stackelberg routing in arbitrary networks. *Math. Oper. Res.* 35(2):330–346.

Cachon G, Lariviere M (1999) Capacity choice and allocation: Strategic behavior and supply chain performance. *Management Sci.* 45(8):1091–1108.

Cachon G, Netessine S (2004) Game-theoretic applications in supply chain analysis. Simchi-Levi D, Wu SD, Shen Z. eds. *Supply Chain Analysis in the eBusiness Era* (Kluwer, Academic Publishers, Dordrecht, The Netherlands), 13–59.

Chen B, Gürel S (2012) Resource allocation games of utilitarian social objectives. *J. Scheduling* 15(2):157–164.

Cominetti R, Correa JR, Stier-Moses NE (2009) The impact of oligopolistic competition in networks. *Oper. Res.* 57(6):1421–1437.

Correa JR, Queyranne M (2010) Efficiency of equilibria in restricted uniform machine scheduling with minsuit social cost. Submitted.

Correa JR, Schulz AS, Stier-Moses NE (2004) Selfish routing in capacitated networks. *Math. Oper. Res.* 29(4):961–976.

Correa JR, Schulz AS, Stier-Moses NE (2007) Fast, fair, and efficient flows in networks. *Oper. Res.* 55(2):215–225.

Czumaj A, Vöcking B (2002) Tight bounds for worst-case equilibria. *ACM-SIAM Sympos. Discrete Algorithms* (*SODA*) (ACM/SIAM, New York), 413–420.

Epstein A, Feldman M, Mansour Y (2007) Strong equilibrium in cost sharing connection games. *ACM Conf. Electronic Commerce* (*ACM-EC*) (ACM, New York).

Even-Dar E, Mansour Y (2005) Fast convergence of selfish rerouting. *Sixteenth ACM-SIAM Sympos. Discrete Algorithms* (*SODA*) (SIAM, Philadelphia), 772–781.

Fabrikant A, Luthra A, Maneva E, Papadimitriou C, Shenker S (2003) On a network creation game. *ACM Sympos. Principles of Distributed Comput.* (*PODC*) (ACM, New York).

Feldman M, Tamir T (2009) Approximate strong equilibrium in job scheduling games. *J. Artificial Intelligence Res.* 36(1):387–414.

Feldman M, Tamir T (2012) Convergence rate of best response dynamics in scheduling games with conflicting congestion effects. Manuscript, Hebrew University of Jerusalem, Jerusalem.

Fiat A, Kaplan H, Levi M, Olonetsky S (2007) Strong price of anarchy for machine load balancing. *ICALP* (Springer, Berlin).

Fotakis D, Mavronicolas M, Kontogiannis S, Spiraklis P (2002) The structure and complexity of Nash equilibria for a selfish routing game. *Internat. Colloquium on Automata, Languages, and Programming* (*ICALP*) (Springer, Berlin), 510–519.

Garey MR, Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman and Company, San Francisco).

Graham RL (1969) Bounds on multiprocessing timing anomalies. *SIAM J. Appl. Math.* 17(2):263–269.

Hakimi SL (1965) Optimal distribution of switching centers in a communication network and some related graph theoretic problems. *Oper. Res.* 13(3):462–475.

Herzog S, Shenker S, Estrin D (1997) Sharing the "cost" of multicast trees: An axiomatic analysis. *IEEE/ACM Trans. Networking* 5(6):847–860.

Jain K, Mahdian M (2007) Cost sharing. Nisan N, Roughgarden T, Tardos E, Vazirani V, eds. *Algorithmic Game Theory* (Cambridge University Press, New York), 385–410.

Koutsoupias E, Papadimitriou C (2009) Worst-case equilibria. *Comput. Sci. Rev.* 3(2):65–69.

Lavi R (2007) Computationally efficient approximation mechanisms. Nisan N, Roughgarden T, Tardos E, Vazirani V, eds. *Algorithmic Game Theory* (Cambridge University Press, New York), 301–330.

Monderer D, Shapley LS (1996) Potential games. *Games Econom. Behav.* 14(1):124–143.

Moulin H, Shenker S (1992) Serial cost sharing. *Econometrica* 60(5):1009–1037.

Moulin H, Shenker S (2001) Strategyproof sharing of submodular costs: Budget balance versus efficiency. *J. Econom. Theory* 18(3):511–533.

Nash J (1951) Non-cooperative games. *Ann. Math.* 54(2):286–295.

Nisan N, Ronen A (2001) Algorithmic mechanism design. *Games Econom. Behav.* 35(1–2):166–196.

Papadimitriou C (2001) Algorithms, games, and the Internet. *Proc.* 33*rd ACM Sympos. Theory Comput.* (*STOC*) (ACM, New York), 749–753.

Parlakturk AK, Kumar S (2004) Self-interested routing in queueing networks. *Management Sci.* 50(7):949–966.

Perakis G (2007) The "price of anarchy" under nonlinear and asymmetric costs. *Math. Oper. Res.* 32(3):614–628.

Perakis G, Roels G (2007) The price of anarchy in supply chains: Quantifying the efficiency of price-only contracts. *Management Sci.* 53(8):1249–1268.

Rawls J (1971) *Theory of Justice* (Harvard University Press, Cambridge, MA).

Rosenthal RW (1973) A class of games possessing pure-strategy Nash equilibria. *Internat. J. Game Theory* 2(1):65–67.

Roughgarden T (2007) Routing games. Nisan N, Roughgarden T, Tardos E, Vazirani V, eds. *Algorithmic Game Theory* (Cambridge University Press, New York), 461–486.

Vetta AR (2002) Nash equilibria in competitive societies with applications to facility location, traffic routing, and auctions. *Sympos. Foundations Comput. Sci.* (*FOCS*) (IEEE Computer Society Press), 416–425.

Vöcking B (2007) Selfish load balancing. Nisan N, Roughgarden T, Tardos E, Vazirani V, eds. *Algorithmic Game Theory* (Cambridge University Press, New York), 517–542.

Michal Feldman is an associate professor at the School of Business Administration at the Hebrew University of Jerusalem. She is a visiting scholar at the School of Engineering and Applied Sciences (SEAS) at Harvard University during 2011–2013 as a Marie Curie Fellow. Her research focuses on the intersection of game theory, computer science, and microeconomics, a field often termed "algorithmic game theory."

Tami Tamir is an associate professor at the School of Computer Science at the Interdisciplinary Center in Israel. Her research interests include design and analysis of algorithms, resource allocation problems, and multimedia-on-demand systems.