Coordination Mechanisms for Selfish Scheduling

Nicole Immorlica [a,1], Li (Erran) Li [b], Vahab S. Mirrokni [c,1], Andreas S. Schulz [d]

[a] *Northwestern University, Evanston, IL*

[b] *Bell Labs, Murray Hill, NJ*

[c] *Google Research, New York, NY*

[d] *Massachusetts Institute of Technology, Cambridge, MA*

Abstract

In machine scheduling, a set of jobs must be scheduled on a set of machines so as to minimize some global objective function, such as the makespan, which we consider in this paper. In practice, jobs are often controlled by independent, selfishly acting agents, which each select a machine for processing that minimizes the (expected) completion time. This scenario can be formalized as a game in which the players are job owners, the strategies are machines, and a player's disutility is the completion time of its jobs in the corresponding schedule. The equilibria of these games may result in larger-than-optimal overall makespan. The price of anarchy is the ratio of the worst-case equilibrium makespan to the optimal makespan. In this paper, we design and analyze scheduling policies, or coordination mechanisms, for machines which aim to minimize the price of anarchy of the corresponding game. We study coordination mechanisms for four classes of multiprocessor machine scheduling problems and derive upper and lower bounds on the price of anarchy of these mechanisms. For several of the proposed mechanisms, we also prove that the system converges to a pure-strategy Nash equilibrium in a linear number of rounds. Finally, we note that our results are applicable to several practical problems arising in communication networks.

Key words: Mechanism design, price of anarchy, scheduling, local search

[1] Part of this work was done while this author was with the Massachusetts Institute of Technology.

1 Introduction

With the advent of the Internet, large-scale autonomous systems have become increasingly common. These systems consist of many independent and selfishly acting agents, all competing to share a common resource such as bandwidth in a network or processing power in a parallel computing environment. Practical settings range from smart routing among stub autonomous systems in the Internet [20] to selfish user association in wireless local area networks [7]. In many systems of this kind, it is infeasible to impose some centralized control on the users. Rather, a centralized authority can only design protocols a priori and hope that the independent and selfish choices of the users—given the rules of the protocol—combine to create socially desirable results.

This approach, termed *mechanism design*, has received considerable attention in the recent literature (see, for example, [30]). Its goal is to design system-wide rules which, given the selfish decisions of the users, maximize the total social welfare. The degree to which these rules approximate the social welfare in a worst-case equilibrium is known as the *price of anarchy* of the mechanism. This concept was introduced in 1999 by Koutsoupias and Papadimitriou [26] in the context of *selfish scheduling games* (see below for details). This seminal paper has spawned a series of results which attempt to design mechanisms with minimum price of anarchy for a variety of games. One approach for achieving this goal is to impose economic incentives on users in the form of *tolls*; i.e., the disutility of a user is affected by a monetary payment to some central authority for the use of a particular strategy such as a route in a network [6,11,17] . Another approach assumes that the central authority is able to enforce particular strategies upon some fraction of the users, which might help to influence the decisions of the other users in the desired way [5,25,31,34].

A drawback of the above approaches is that many of the known algorithms assume global knowledge of the system and thus have high communication complexity. In many settings, it is important to be able to compute mechanisms locally. A third approach, which we follow here, is called *coordination mechanisms*, first introduced by Christodoulou, Koutsoupias, and Nanavati [10]. A coordination mechanism is a *local* policy that assigns a cost to each strategy s, where the cost of s is a function of the agents that have chosen s. Consider, for example, a *selfish scheduling game* in which there are n jobs owned by independent agents, m machines, and a processing time p_{ij} for job i on machine j. Each agent selects a machine on which to schedule its job with the objective of minimizing its own completion time. The social objective is to minimize the maximum completion time. A coordination mechanism for this game is a local policy, one for each machine, that determines how to schedule jobs assigned to that machine. It is important to emphasize that a machine's policy is a function only of the jobs assigned to that machine. This allows the

policy to be implemented in a completely distributed fashion.

Coordination mechanisms are closely related to local search algorithms. A local search algorithm iteratively selects a solution "close" to the current solution that improves the global objective. It selects the new solution from among those within some *search neighborhood* of the current solution. Given a coordination mechanism, we can define a local search algorithm whose search neighborhood is the set of best responses for each agent. Similarly, given a local search algorithm, it is sometimes possible to define a coordination mechanism whose pure-strategy equilibria are local optima with respect to the search neighborhood. The locality gap of the search neighborhood, i.e., the approximation factor of the local search algorithm, is precisely the price of anarchy of the corresponding coordination mechanism, and vice versa. In particular, designing new coordination mechanisms may lead to the discovery of new local search algorithms.

In this paper, we are primarily interested in the properties of *pure-strategy Nash equilibria* for the selfish scheduling games we consider. A pure-strategy Nash equilibrium is an assignment of jobs to machines such that no job has a unilateral incentive to switch to another machine. Although a non-cooperative game always has a mixed-strategy equilibrium [29], it may in general not have a pure-strategy equilibrium. However, pure-strategy Nash equilibria are arguably more natural and, when they exist, they may better predict game play. We also bound the *rate of convergence* of the mechanism. That is, if the jobs, starting from an arbitrary solution, iteratively play their best-response strategies, how long does it take for the mechanism to reach a Nash equilibrium? [2] Guarantees of this sort are important for the applicability of bounded price-of-anarchy mechanisms in a practical setting.

Preliminaries. In parallel machine scheduling, n jobs must be processed on m machines. Job i has processing time p_{ij} on machine j. A schedule μ is a function mapping each job to a machine. The makespan of a machine j in schedule μ is $M_j = \sum_{i:j=\mu(i)} p_{ij}$. The goal is to find a schedule μ which minimizes the maximum makespan $C_{\max} = \max_j M_j$. Different assumptions regarding the relationship between processing times yield different scheduling problems, of which we consider the following four: (i) *identical* machine scheduling ($P|\ |C_{\max}$) in which $p_{ij} = p_{ik} = p_i$ for each job i and machines j and k; (ii) *uniform* or *related* machine scheduling ($Q|\ |C_{\max}$) in which $p_{ij} = p_i/s_j$, where p_i is the processing requirement of job i and s_j is the speed of machine j; (iii) machine scheduling for *restricted assignment* or *bipartite* machine scheduling ($B|\ |C_{\max}$) in which each job i can be scheduled on a subset S_i of

[2] In this paper, we use the term Nash equilibrium for pure-strategy Nash equilibrium.

machines only, i.e., p_{ij} is equal to p_i if $j \in S_i$ and is equal to ∞ otherwise; and (iv) *unrelated* machine scheduling ($R|\ |C_{\max}$) in which the processing times p_{ij} are arbitrary positive numbers.

In *selfish scheduling*, each job is owned by an independent agent whose goal is to minimize the completion time of its own job. To induce these selfish agents to take globally near-optimal actions, we consider the notion of a *coordination mechanism* [10]. A coordination mechanism is a set of *scheduling policies*, one for each machine. A scheduling policy for machine j takes as input a set of jobs on machine j along with their processing times on machine j and outputs an ordering in which they will be scheduled. The policy is run locally at each machine, and so it does not have access to information regarding the global state of the system (e.g., the set of all jobs). A coordination mechanism defines a game with n agents. An agent's strategy set is the set of possible machines $\{1, ..., m\}$. Given a strategy profile, the disutility of agent i is the (expected) completion time of job i in the schedule defined by the coordination mechanism. We study four coordination mechanisms. In the ShortestFirst and LongestFirst policies, we sequence the jobs in non-decreasing and non-increasing order of their processing times, respectively. In the Randomized policy, we process the jobs in random order. [3] In the Makespan policy, we process all jobs on the same machine in parallel, and so the completion time of a job on machine j is the makespan M_j of machine j.

We are interested in the properties of the solutions resulting from the agents' selfish behavior, in particular, in the *price of anarchy*. In our setting, the price of anarchy is the worst-case ratio (over all instances) of the maximum makespan in a Nash equilibrium to the optimal makespan. For each policy and each scheduling problem, we prove upper and lower bounds on the price of anarchy. Some of these bounds are already known as the approximation factor of some local search algorithms or the price of anarchy in some selfish load balancing games. All bounds, known and new, are summarized in Table 1.

In order to avoid subtle difficulties arising from ties between processing times of jobs, we assume that all deterministic policies (especially ShortestFirst and LongestFirst) and greedy algorithms resolve ties in a consistent manner. Moreover, we require the tie-breaking rule to be deterministic and independent of irrelevant alternatives; i.e., the resolution of a tie between jobs i and i' is not affected by the presence or absence of job i''. One such tie-breaking rule is the *alphabetically first* rule. When there is a tie between the processing times of two jobs, the alphabetically first rule always chooses the one with the smaller identifier. In fact, we may assume w.l.o.g. that we always use the alphabetically first rule because we can easily relabel jobs such that in a tie job i is considered before i' whenever $i < i'$.

[3] The Randomized policy is also known as the batch model [26].

	Makespan	ShortestFirst	LongestFirst	Randomized
$P\|\ \|C_{\max}$	$2-\frac{2}{m+1}$ [16,32]	$2-\frac{1}{m}$ [21]*	$\frac{4}{3}-\frac{1}{3m}$ [22,10]	$2-\frac{2}{m+1}$ [16,32]
$Q\|\ \|C_{\max}$	$\Theta(\frac{\log m}{\log\log m})$ [12]	$\Theta(\log m)$ [1], *	$1.52 \le P \le 1.59$ [14,18]*	$\Theta(\frac{\log m}{\log\log m})$ [12]
$B\|\ \|C_{\max}$	$\Theta(\frac{\log m}{\log\log m})$ [19,2]	$\Theta(\log m)$ [1], *	$\Theta(\log m)$ [4], *	$\Theta(\frac{\log m}{\log\log m})$ [19,2]
$R\|\ \|C_{\max}$	Unbounded [32]	$\log m \le P \le m$ [23,9]*	Unbounded	$\Theta(m)$ *

Table 1
The price of anarchy for four different policies and scheduling problems. All upper and lower bounds hold for pure Nash equilibria. The upper bounds of the Randomized policy for $R|\ |C_{\max}$ and $Q|\ |C_{\max}$ are valid for the maximum of the expected load on any machine in mixed Nash equilibria. The results marked by * are proved in this paper or follow from the correspondence of certain greedy schedules with equilibrium schedules, established in this paper.

Our Contribution. We present upper and lower bounds on the price of anarchy for the Randomized, the ShortestFirst, and the LongestFirst policies. We give a proof that the price of anarchy of any deterministic coordination mechanism (including ShortestFirst and LongestFirst) for $Q|\ |C_{\max}$ and $B|\ |C_{\max}$ is $O(\log m)$. This result is also implied by earlier results on greedy algorithms for these scheduling problems [1,4] because we show that—in certain situations—a schedule is a Nash equilibrium if and only if it can be produced by these algorithms. We also prove that the price of anarchy of any deterministic coordination mechanism (such as ShortestFirst and LongestFirst) is $\Omega(\log m)$ for $B|\ |C_{\max}$. In addition, we analyze the Randomized policy for $R|\ |C_{\max}$ and prove a bound of $\Theta(m)$ on its price of anarchy. We further study the convergence to and the existence of pure Nash equilibria for the ShortestFirst and LongestFirst policies. In particular, we show that the mechanism based on the ShortestFirst policy for $R|\ |C_{\max}$ is a potential game, and any sequence of best responses converges to a Nash equilibrium after at most n rounds. We also prove fast convergence of the LongestFirst policy for $Q|\ |C_{\max}$ and $B|\ |C_{\max}$.

Related work. The Makespan policy is perhaps the best known policy among the above policies; see, e.g., [2,12,19,26]. Czumaj and Vöcking [12] gave tight results on the price of anarchy for the Makespan policy for mixed Nash equilibria and $Q|\ |C_{\max}$. Gairing et al. [19] and Awerbuch et al. [2] obtained a tight bound for the price of anarchy of the Makespan policy for pure Nash equilibria and $B|\ |C_{\max}$. In addition, Gairing et al. [19] presented a polynomial-time algorithm for computing a pure Nash equilibrium with makespan at most twice the optimal makespan.

Coordination mechanism design was introduced by Christodoulou, Koutsoupias, and Nanavati [10]. In their paper, they analyzed the LongestFirst policy for $P|\ |C_{\max}$. As we have mentioned before, the price of anarchy for coor-

dination mechanisms is closely related to the approximation factor of local search algorithms. Vredeveld surveyed some of the results on local search algorithms for scheduling problems in his thesis [35]. The "jump model" described therein (see also [32]) is similar to the Makespan policy. (More precisely, any Nash equilibrium under that policy is a jump-optimal schedule, but not necessarily vice versa.) The "push model" is related to the LongestFirst policy. Cho and Sahni [9] showed that the approximation factor of the shortest-first greedy algorithm is not better than $\log m$ for $Q|\ |C_{\max}$. Aspnes et al. [1] and Azar et al. [4] proved that the greedy list scheduling algorithm is an $O(\log m)$-approximation algorithm for $B|\ |C_{\max}$ and $Q|\ |C_{\max}$, respectively. Their proofs can be used to bound the price of anarchy of the LongestFirst and ShortestFirst polices for $B|\ |C_{\max}$ and $Q|\ |C_{\max}$. Ibarra and Kim [23] proved that the shortest-first greedy algorithm is an m-approximation for $R|\ |C_{\max}$. Our lower bound example for the ShortestFirst and the LongestFirst policy is the same as an example in [13]. Davis and Jaffe [13] also gave a $\sqrt{m}$-approximation for $R|\ |C_{\max}$. The best known approximation factor for $R|\ |C_{\max}$ is obtained by a 2-approximation algorithm due to Lenstra, Shmoys, and Tardos [27].

Even-Dar et al. [15] considered the convergence time to Nash equilibria for variants of the selfish scheduling problem. In particular, they studied the Makespan policy and bounded the number of required steps to reach a pure Nash equilibrium.

After the appearance of the conference version of this paper [24], Azar, Jain, and Mirrokni [3] improved on some of the results presented here. In particular, they provided a matching lower bound of $\Omega(m)$ for the price of anarchy of the ShortestFirst policy and $R|\ |C_{\max}$. Moreover, they gave an ordering policy whose price of anarchy is $\Theta(\log m)$ for $R|\ |C_{\max}$. However, this policy needs to know the minimum processing time of a job on any machine to compute the schedule for a given machine. Hence, their mechanism is not local, strictly speaking.[4] However, they also showed that no deterministic local ordering policy can achieve a price of anarchy better than $\Omega(m)$ for $R|\ |C_{\max}$.

The paper is organized as follows. In Section 2, we prove upper bounds on the price of anarchy, for various coordination mechanisms and scheduling problems. Corresponding lower bounds are discussed in Section 3. Afterwards, we study the time it takes to reach a Nash equilibrium (Section 4). Section 5 contains our concluding remarks.

[4] In contrast to the terminology used here, Azar et al. call policies that only make use of the processing times of jobs on the same machine, *strongly* local, and policies that consider all characteristics of the jobs to be scheduled on the same machine (and not any information about jobs on other machines), local.

2 Upper Bounds on the Price of Anarchy

2.1 The ShortestFirst *Policy*

We begin by bounding the price of anarchy of the ShortestFirst policy for $R|\ |C_{\max}$. We note that there is a direct correspondence between outputs of the well-known shortest-first greedy algorithm for machine scheduling [23, Algorithm D] and Nash equilibria of the ShortestFirst policy.

Theorem 1 *The set of Nash equilibria for the* ShortestFirst *policy and* $R|\ |C_{\max}$ *is precisely the set of solutions that can be generated by the shortest-first greedy algorithm.*

PROOF. Consider a schedule μ produced by the shortest-first greedy algorithm. On each machine, jobs are scheduled in order of non-decreasing processing times. That is, the ShortestFirst policy is in effect. Suppose it would be beneficial for job i to switch from its current machine $\mu(i)$ to another machine; let j be the machine on which i would complete first. Then the greedy algorithm would also have scheduled i on j by the shortest-first rule, a contradiction.

We prove the other direction by induction on the number of jobs. For $n = 1$, there is nothing to prove. So let μ be an equilibrium schedule with $n+1$ jobs. Let i be a job that determines the makespan; i.e., the completion time of i equals $C_{\max}$. If we remove i from μ, the remaining schedule, let's call it μ', is still a Nash equilibrium because i is last. Hence, μ' can be constructed by using the shortest-first greedy algorithm. Because μ is a Nash equilibrium, i has no incentive to switch to another machine. In particular, the processing time of i on each machine j is at least as large as that of any other job currently assigned to j. It follows that μ can also be interpreted as an output of the shortest-first greedy algorithm. □

The implication is that any bound on the approximation factor of the shortest-first greedy algorithm is also a bound on the price of anarchy of the ShortestFirst policy for $R|\ |C_{\max}$. In particular, Theorem 1 and a result of Ibarra and Kim [23, Theorem 1] prove that the price of anarchy of ShortestFirst for $R|\ |C_{\max}$ is at most m. We include a proof for the sake of completeness.

Theorem 2 *The price of anarchy of the* ShortestFirst *policy for* $R|\ |C_{\max}$ *is at most* m.

PROOF. Fix any Nash equilibrium μ and label the jobs in non-decreasing order of completion times, which, according to Theorem 1, coincides with the order in which they are scheduled in the shortest-first greedy algorithm. Let M^i be the makespan of the schedule μ restricted to jobs 1 through i. Let $p_i = \min_j p_{ij}$ be the shortest possible processing time of job i. It follows that $M^i \leq M^{i-1} + p_i$. Using $\frac{1}{m}\sum_{i=1}^{n} p_i$ as a lower bound on the cost of an optimal solution, we see that the social cost of the Nash equilibrium is $M^n = \sum_{i=1}^{n}(M^i - M^{i-1}) \leq \sum_{i=1}^{n} p_i$, and thus is at most m times the cost of an optimal solution. □

We can further prove that the price of anarchy of the ShortestFirst policy for $Q|\ |C_{\max}$ is $\Theta(\log m)$. This also shows that the approximation factor of the shortest-first greedy algorithm for $Q|\ |C_{\max}$ is $\Theta(\log m)$, as was previously observed by Aspnes et al. [1]. In fact, the bound on the price of anarchy can be derived from their result as well. Our proof of the upper bound, derived independently, uses ideas from the proof of the price of anarchy for the Makespan policy for $Q|\ |C_{\max}$ by Czumaj and Vöcking [12]. The lower bound follows from earlier work by Cho and Sahni [9].

We prove the upper bound for any *deterministic* coordination mechanism. A coordination mechanism is deterministic if the scheduling policies do not use randomization to determine the schedules. We prove that the price of anarchy for deterministic mechanisms, including the ShortestFirst policy, for $Q|\ |C_{\max}$ is $O(\log m)$.

Theorem 3 *The price of anarchy of any deterministic policy for* $Q|\ |C_{\max}$ *is* $O(\log m)$. *In particular, the price of anarchy of the* ShortestFirst *policy for* $Q|\ |C_{\max}$ *is* $O(\log m)$.

PROOF. Assume that $s_1 \geq s_2 \geq \cdots \geq s_m$. Let ω be an optimal schedule with makespan OPT. Let μ be a Nash equilibrium with makespan $k\mathrm{OPT} \leq C < (k+1)\mathrm{OPT}$, for some integer k. Let M_j be the makespan of machine j in μ.

Let m_ℓ $(1 \leq \ell \leq k-1)$ be the minimum index of a machine such that the makespan of machine $m_\ell + 1$ is less than $(k-\ell)\mathrm{OPT}$, i.e., for any $j \leq m_\ell$, $M_j \geq (k-\ell)\mathrm{OPT}$, and $M_{m_\ell+1} < (k-\ell)\mathrm{OPT}$. First, we prove that $M_1 \geq (k-1)\mathrm{OPT}$ and thus $m_1 \geq 1$. Suppose that $M_1 < (k-1)\mathrm{OPT}$. Let i be the last job on the machine with the maximum makespan. Thus, i's completion time is $C \geq k\mathrm{OPT}$. It is clear that $p_i/s_1 \leq \mathrm{OPT}$ as machine 1 is the fastest machine. This shows that job i has an incentive to switch to machine 1, since its completion time on machine 1 is strictly less than $(k-1)\mathrm{OPT} + p_i/s_1 \leq k\mathrm{OPT}$. This contradicts the fact that μ is a Nash equilibrium. Therefore, $M_1 \geq (k-1)\mathrm{OPT}$ and $m_1 \geq 1$.

Next, we prove that $m_\ell \geq (\ell - t - 1)m_t$ for any $1 \leq t < \ell \leq k-1$. Let W be the set of all jobs with completion times greater than or equal to $(k-\ell+1)\mathrm{OPT}$ that are scheduled on machines $1, 2, \ldots, m_t$ in μ. Consider a job i in W. We claim that job i is scheduled on one of the machines $1, 2, \ldots, m_\ell$ in the optimal solution ω. If job i is scheduled on machine $j > m_\ell$ in ω, then $p_i/s_{m_\ell+1} \leq p_i/s_j \leq \mathrm{OPT}$, thus job i has an incentive to switch to machine $m_\ell + 1$ as its completion time on machine $m_\ell + 1$ is less than $(k-\ell)\mathrm{OPT} + p_i/s_{m_\ell+1} \leq (k - \ell + 1)\mathrm{OPT}$. Therefore, all jobs in W are scheduled on machines $1, 2, \ldots, m_\ell$ in ω. The sum of processing times of jobs in W on machine $q \leq m_t$ in μ is at least $((k-t)-(k-\ell+1))\mathrm{OPT}s_q = (\ell - t - 1)\mathrm{OPT}s_q$. Let w be the sum of processing times of the jobs in W. Thus, $w \geq (\ell - t - 1)\mathrm{OPT}\sum_{q=1}^{m_t} s_q$. All this load is on machines $1, 2, \ldots, m_\ell$ in the optimal solution ω, thus the total processing time is at most $w \leq \mathrm{OPT}\sum_{q=1}^{m_\ell} s_q$. Therefore, $\sum_{q=1}^{m_\ell} s_q \geq (\ell - t - 1)\sum_{q=1}^{m_t} s_q$. Since the machines are indexed in non-increasing order of their speeds, we get $m_\ell \geq (\ell - t - 1)m_t$.

In particular, $m_{k-1} \geq 2m_{k-4} \geq 2^i \cdot m_{k-3i-1} \geq 2^{\frac{k-4}{3}} m_1$. Using $m \geq m_{k-1}$ and $m_1 \geq 1$, we get $k = O(\log m)$, as required. □

In addition, we can prove that the price of anarchy of any deterministic mechanism for $B|\ |C_{\max}$ is $\Theta(\log m)$. For the ShortestFirst policy, the corresponding upper bound is implied by a result of Azar et al. [4] on the approximation factor of the greedy algorithm for $B|\ |C_{\max}$, but our proof is independent of theirs and uses the ideas of the proof for the Makespan policy by Gairing et al. [19]. The lower bound is discussed in Section 3.

Theorem 4 *The price of anarchy of any deterministic policy for* $B|\ |C_{\max}$ *is* $O(\log m)$. *In particular, the price of anarchy of the* ShortestFirst *policy for* $B|\ |C_{\max}$ *is* $O(\log m)$.

PROOF. Let μ be a Nash equilibrium. We also fix an optimal schedule. Let OPT be its value. Let C be the makespan of μ, and assume that $k\mathrm{OPT} \leq C < (k+1)\mathrm{OPT}$. Moreover, let M_j be the makespan of machine j in μ. Let $\mathcal{M}_0$ be the set of machines with makespan greater than $k\mathrm{OPT}$. We exhibit a family of disjoint sets of machines $\mathcal{M}_0, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5, \ldots, \mathcal{M}_{k-1}$ such that (i) for all $3 \leq \ell \leq k-1$, $|\mathcal{M}_\ell| \geq (\ell-2)|\mathcal{M}_0| + (\ell-5)|\mathcal{M}_3| + (\ell-6)|\mathcal{M}_4| + \cdots + 2|\mathcal{M}_{\ell-4}| + |\mathcal{M}_{\ell-3}| - |\mathcal{M}_{\ell-1}|$, and (ii) for all $3 \leq \ell \leq k-1$ and $\ell = 0$, for each $j \in \mathcal{M}_\ell$, $M_j \geq (k-\ell)\mathrm{OPT}$.

We construct this family inductively as follows. We first prove that there exists a set $\mathcal{M}_3$ of machines, disjoint from $\mathcal{M}_0$, such that the makespan of all jobs in $\mathcal{M}_3$ is at least $(k-3)\mathrm{OPT}$ and $|\mathcal{M}_3| \geq |\mathcal{M}_0|$. Consider all jobs that finish after time $(k-2)\mathrm{OPT}$ on machines in $\mathcal{M}_0$. The total processing

time of these jobs is at least $2|\mathcal{M}_0|$OPT. At most $|\mathcal{M}_0|$OPT of it is scheduled on the same machines in the optimal solution, thus at least $|\mathcal{M}_0|$OPT of the total processing time of these jobs is scheduled on other machines in the optimum solution. The number of such machines is at least $|\mathcal{M}_0|$. Call this set $\mathcal{M}_3$. Thus, $\mathcal{M}_3 \cap \mathcal{M}_0 = \emptyset$, and $|\mathcal{M}_3| \geq |\mathcal{M}_0|$. For any machine $j \in \mathcal{M}_3$, $M_j \geq (k-3)$OPT, since otherwise a job i on a machine in $\mathcal{M}_0$ that finishes after $(k-2)$OPT could switch to machine j and its completion time would be less than $(k-3)\text{OPT} + p_i \leq (k-2)\text{OPT}$. This contradicts the fact that job i finishes after $(k-2)$OPT in μ and μ is a Nash equilibrium. This proves the desired property about jobs scheduled on machines in $\mathcal{M}_3$.

Now, assuming the induction hypothesis is true for $\mathcal{M}_0, \mathcal{M}_3, \mathcal{M}_4, \ldots, \mathcal{M}_{\ell-1}$, we prove the existence of $\mathcal{M}_\ell$. Consider the total processing requirement of jobs on machines in $\mathcal{M}_0 \cup \mathcal{M}_3 \cup \mathcal{M}_4 \cup \ldots \cup \mathcal{M}_{\ell-1}$ that finish after time $(k-\ell+1)$OPT in μ. The total processing time of jobs that complete after $(k-\ell+1)$OPT on a machine $j \in \mathcal{M}_q$ is at least $(\ell-1-q)$OPT, for $q = 0, 3, 4, \ldots, \ell-1$. Therefore, the total processing time of jobs that complete after time $(k-\ell+1)$OPT in μ is at least $(\ell-1)|\mathcal{M}_0|\text{OPT} + (\ell-4)|\mathcal{M}_3|\text{OPT} + (\ell-5)|\mathcal{M}_4|\text{OPT} + \cdots + |\mathcal{M}_{\ell-2}|\text{OPT}$. At most $\text{OPT}(|\mathcal{M}_0| + |\mathcal{M}_3| + |\mathcal{M}_4| + \cdots + |\mathcal{M}_{\ell-1}|)$ of this amount is scheduled on machines in $\mathcal{M}_0 \cup \mathcal{M}_3 \cup \mathcal{M}_4 \cup \ldots \cup \mathcal{M}_{\ell-1}$ in the optimum solution. Hence, at least $(\ell-2)|\mathcal{M}_0|\text{OPT} + (\ell-5)|\mathcal{M}_3|\text{OPT} + (\ell-6)|\mathcal{M}_4|\text{OPT} + \cdots + |\mathcal{M}_{\ell-3}|\text{OPT} - |\mathcal{M}_{\ell-1}|\text{OPT}$ of it is scheduled on some other machines in the optimum solution. Call this set of machines $\mathcal{M}_\ell$. $\mathcal{M}_\ell$ is disjoint from all sets of machines $\mathcal{M}_i$, for $i < \ell$ and $|\mathcal{M}_\ell| \geq (\ell-2)|\mathcal{M}_0| + (\ell-5)|\mathcal{M}_3| + (\ell-6)|\mathcal{M}_4| + \ldots + |\mathcal{M}_{\ell-3}| - |\mathcal{M}_{\ell-1}|$, as desired. Moreover, the makespan of a machine j in $\mathcal{M}_\ell$ is at least $(k-\ell)$OPT. Otherwise, a job i that finishes after $(k-\ell+1)$OPT on $\mathcal{M}_0 \cup \mathcal{M}_3 \cup \mathcal{M}_4 \cup \ldots \cup \mathcal{M}_{\ell-1}$ could switch from its current machine to machine j in $\mathcal{M}_\ell$ and its completion time is less than $(k-\ell)\text{OPT} + p_i \leq (k-\ell+1)\text{OPT}$. This contradicts the fact that job i is scheduled after $(k-\ell+1)$OPT in μ and μ is a Nash equilibrium. This completes the proof of the induction step.

We know that $m \geq |\mathcal{M}_0| + \sum_{\ell=3}^{k-1} |\mathcal{M}_\ell|$. Let $b_0 = |\mathcal{M}_0|$ and $b_\ell = (\ell-2)b_0 + (\ell-5)b_3 + (\ell-6)b_4 + \cdots + b_{\ell-3} - b_{\ell-1}$ for all $\ell \geq 3$. By defining $a_\ell = b_0 + \sum_{i=3}^{\ell} b_i$, we get $b_\ell = a_\ell - a_{\ell-1}$. Also, using the above recurrence relation for b_ℓ, we can easily prove that $b_\ell - b_{\ell-1} = b_0 + \sum_{i=3}^{\ell-2} b_i - b_{\ell-1} = a_{\ell-2} - b_{\ell-1}$. Thus, $a_{\ell-2} = b_\ell = a_\ell - a_{\ell-1}$. As a result, $a_\ell = a_{\ell-1} + a_{\ell-2}$ for $\ell \geq 3$, and by definition, $a_2 = a_1 = a_0 = b_0 \geq 1$. Hence, a_ℓ is the product of the $(\ell+1)$-st Fibonacci number and b_0, $\ell \geq 1$, and therefore, $a_\ell \geq \phi^{\ell-1}$, where ϕ is the golden ratio. Thus, $m \geq a_{k-1} \geq \phi^{k-2}$, which implies $k = O(\log m)$. $\square$

2.2 The LongestFirst *Policy*

It is easy to see that the price of anarchy of the LongestFirst policy for unrelated parallel machine scheduling is unbounded. It is known that the price of anarchy of this policy for $P|\ |C_{\max}$ is bounded from above by $\frac{4}{3} - \frac{1}{3m}$, and this is tight [10]. Theorems 3 and 4 show that the price of anarchy of the LongestFirst policy for $B|\ |C_{\max}$ and $Q|\ |C_{\max}$ is $O(\log m)$. In Section 3, we prove that this bound is tight for $B|\ |C_{\max}$. For $Q|\ |C_{\max}$ however, the price of anarchy is bounded by a constant. This follows from earlier work by Dobson [14] and Friesen [18] on the longest-first greedy algorithm and the following theorem.

Theorem 5 *For $P|\ |C_{\max}$, $Q|\ |C_{\max}$, and $B|\ |C_{\max}$, the set of Nash equilibria for the* LongestFirst *policy is precisely the set of solutions that can be returned by the longest-first greedy algorithm.*

PROOF. The proof is similar to that of Theorem 1. In a schedule returned by the greedy algorithm, the jobs are scheduled in non-increasing order of processing times on each machine. Moreover, no job has an incentive to switch because otherwise the greedy algorithm would have done the same. For the other direction, we again use induction on the number of jobs. This time, we remove the job with the shortest processing time from the equilibrium schedule. The claim follows easily. □

2.3 The Randomized *Policy*

In the Randomized policy, an agent's disutility is the *expected* completion time of its job. We begin by characterizing the condition under which an agent has an incentive to change strategies. Consider a job i on machine j and let S_j be the set of jobs assigned to machine j. Then the disutility of agent i under the Randomized policy is:

$$p_{ij} + \frac{1}{2} \sum_{i' \neq i, i' \in S_j} p_{i'j}.$$

Letting M_j be the makespan of machine j, we see that a job i on machine j has an incentive to change to machine k if and only if:

$$p_{ij} + M_j > 2p_{ik} + M_k.$$

Because of this observation, the randomized policy is the same as the Makespan policy for $P|\ |C_{\max}$ and $B|\ |C_{\max}$. This implies a price of anarchy of at most $2 - \frac{2}{m+1}$ and $O(\frac{\log m}{\log \log m})$ for these settings, respectively.

Here, we bound the price of anarchy of the Randomized policy for $Q|\ |C_{\max}$ and $R|\ |C_{\max}$. In fact, we prove that, in contrast to the Makespan policy, the price of anarchy of the Randomized policy for $R|\ |C_{\max}$ is not unbounded.

Theorem 6 *The price of anarchy of the* Randomized *policy for* $R|\ |C_{\max}$ *is at most* $2m-1$.

PROOF. Let μ be any Nash equilibrium, and let ω be an optimal solution. We consider two groups of jobs: those that are on different machines in ω and μ, and those that are on the same machine. Define S_{qj} as the set of jobs on machine q in μ that are on machine j in ω, and let $L_q = \sum_{i \in \cup_{j \neq q} S_{qj}} p_{iq}$, $O_q = \sum_{i \in \cup_{j \neq q} S_{jq}} p_{iq}$, and $R_q = \sum_{i \in S_{qq}} p_{iq}$. Thus, the makespan of a machine ℓ in μ is $L_\ell + R_\ell$, and the makespan of ℓ in ω is $O_\ell + R_\ell$. Since μ is a Nash equilibrium, for all jobs $i \in S_{qj}$,

$$L_q + R_q + p_{iq} \leq L_j + R_j + 2p_{ij}.$$

Suppose the makespan of μ is achieved on machine ℓ, and the makespan of ω is achieved on machine ℓ'. Then

$$\begin{aligned}
|\cup_{j \neq \ell} S_{\ell j}|(L_\ell + R_\ell) + L_\ell &= \sum_{j \neq \ell} \sum_{i \in S_{\ell j}} (L_\ell + R_\ell + p_{i\ell}) \\
&\leq \sum_{j \neq \ell} \sum_{i \in S_{\ell j}} (L_j + R_j + 2p_{ij}) \\
&\leq |\cup_{j \neq \ell} S_{\ell j}|(L_\ell + R_\ell) + 2 \sum_{j \neq \ell} \sum_{i \in S_{\ell j}} p_{ij} \\
&\leq |\cup_{j \neq \ell} S_{\ell j}|(L_\ell + R_\ell) + 2 \sum_{j \neq \ell} O_j \\
&\leq |\cup_{j \neq \ell} S_{\ell j}|(L_\ell + R_\ell) + 2(m-1)(O_{\ell'} + R_{\ell'}).
\end{aligned}$$

Therefore, the value of the Nash solution μ is at most $2(m-1)(O_{\ell'}+R_{\ell'})+R_\ell \leq (2m-1)(O_{\ell'} + R_{\ell'})$, and so the price of anarchy is at most $2m-1$. $\square$

Unfortunately, we do not know if Nash equilibria exist for the Randomized policy for $R|\ |C_{\max}$, and so the above theorem might be vacuous for some instances. However, we can extend the above proof to bound the maximum of the expected load of a machine in a mixed Nash equilibrium of the Randomized policy for $R|\ |C_{\max}$. If M_j is the expected load of machine j in a mixed Nash equilibrium, then it is easy to show that if the probability of assigning job i to machine q is nonzero, then for any other machine j, $M_q + p_{iq} \leq M_j + 2p_{ij}$. Now, we can define L_q as the expected load of jobs with positive probability

on machine q that are scheduled on machines other than q in the optimum solution. Similar inequalities hold in this setting. This bounds the maximum of the expected load of any machine in a mixed Nash equilibrium. Note that this analysis does not hold for the expected value of the maximum load (see [12] for the difference between these two objective functions). We can further prove that this bound is tight, up to a constant factor (see Theorem 8).

3 Lower Bounds on the Price of Anarchy

In this section, we prove lower bounds on the price of anarchy for coordination mechanisms. Our first result shows that the price of anarchy of a general class of coordination mechanisms for $B|\ |C_{\max}$ and, thus, $R|\ |C_{\max}$ is at least $\log m$. This is interesting in light of the fact that constant-factor LP-based approximation algorithms are known for $R|\ |C_{\max}$ [27], and suggests that it may be hard to obtain similar approximations with local search algorithms.

The example in our proof was used by Davis and Jaffe [13] to show that the approximation factor of the shortest-first greedy algorithm is at least $\log m$.

Theorem 7 *The price of anarchy of any deterministic coordination mechanism is at least* $\log m$ *for* $B|\ |C_{\max}$ *and* $R|\ |C_{\max}$.

PROOF. Consider the following instance of $B|\ |C_{\max}$: there are m jobs and m machines and the processing time of job i on machines $1, 2, \ldots, m-i+1$ is 1. Job i cannot be scheduled on machines $m-i+2, \ldots, m$. Assume that $m = 2^k$, for some positive integer k.

Consider an assignment of jobs to machines as follows. Jobs 1 to $2^{k-1} = m/2$ are assigned to machines 1 to $m/2$ respectively. Jobs $m/2+1$ to $3m/4$ are assigned to machines 1 to $m/4$, respectively. Jobs $3m/4+1$ to $7m/8$ are assigned to machines 1 to $m/8$, and so on. It is not hard to check that this is a Nash equilibrium. The makespan of this assignment is $\log m + 1$. In an optimal assignment, job i is assigned to machine $m-i+1$. Thus the optimal makespan is 1, and so the price of anarchy is at least $\log m$. □

The example in the above proof can be easily changed to show that for $R|\ |C_{\max}$, the price of anarchy of the LongestFirst and ShortestFirst policies is at least $\log m$, even if there is no tie among the processing times.

Theorem 7 proves that if a coordination mechanism is deterministic and policies on different machines are the same, then we cannot hope to get a factor better than $\log m$ for $R|\ |C_{\max}$. One might hope that the Randomized policy

can achieve a constant price of anarchy. However, we have the following lower bound for the Randomized policy.

Theorem 8 *The price of anarchy of the* Randomized *policy for* $R|\ |C_{\max}$ *is at least* $m-1$.

PROOF. Consider a scenario with m machines and $(m-1)^2$ jobs. Split the first $(m-1)(m-2)$ jobs into $m-1$ groups $J_1, \ldots, J_{m-1}$, each of size $m-2$ jobs. For jobs $i \in J_k$, let $p_{ik} = 1$, $p_{im} = 1/m^2$, and $p_{ij} = \infty$ for all other machines j. Form a matching between the remaining $m-1$ jobs and the first $m-1$ machines. Whenever job i is matched to machine j in this matching, set $p_{ij} = 1$ and $p_{im} = 1$. Job i cannot be processed on any other machine.

An optimal solution has makespan 1 and assigns all jobs in $J_1, \ldots, J_{m-1}$ to the last machine and each of the remaining $m-1$ jobs to its corresponding machine in the matching. However, the solution which assigns all jobs in J_k to machine k for all $1 \leq k \leq m-1$ and all remaining $m-1$ jobs to machine m is a Nash equilibrium with makespan $m-1$. To see that this is a Nash equilibrium, consider a job $i \in J_k$. Its disutility on machine k is $\frac{1}{2}(m-3)+1$ while its disutility, if it moved to machine m, would increase to $\frac{1}{2}(m-1)+1/m^2$. Therefore all jobs in $J_1, \ldots, J_{m-1}$ are playing a best response to the current set of strategies. Now consider one of the remaining $m-1$ jobs i. Say job i is matched to machine j in the matching. Then the disutility of job i on machine m is $\frac{1}{2}(m-2)+1$ while its disutility, if it moved to machine j, would remain $\frac{1}{2}(m-2)+1$. Since these jobs are also playing a (weakly) best response to the current set of strategies, the above scenario is a Nash equilibrium in the Randomized policy. □

4 Convergence to Pure-Strategy Nash Equilibria

In practice, it is undesirable if the job to machine mapping keeps changing. The system performance can be adversely affected if players keep reacting to one another's changes of strategies. A good coordination mechanism is one with a small price of anarchy and fast convergence to a Nash equilibrium. In this section, we investigate the convergence of players' selfish behavior. We prove that, except for the case of the Randomized and LongestFirst policies for $R|\ |C_{\max}$ and the Randomized policy for $Q|\ |C_{\max}$, the selfish behavior of players converges to a Nash equilibrium.

We first define the notion of a state graph and a potential function. Let A_i be the set of actions of player i, $i = 1, 2, \ldots, n$. In our setting, each A_i equals the set of machines $\{1, 2, \ldots, m\}$. A state graph $G = (V, E)$ is a directed

graph where the set V of nodes equals $A_1 \times A_2 \times \cdots \times A_n$, and an edge labeled with i exists from state u to v if the only difference between the two states is the action of player i and i's payoff is strictly less in u. A Nash equilibrium corresponds to a node with no outgoing edges. A potential function is a function f mapping the set of states to a totally ordered set such that $f(v)$ is strictly less than $f(u)$ for all edges $uv \in G$. In other words, whenever a player in state u changes its action and improves its payoff, the resulting state v satisfies $f(u) > f(v)$. Note that the existence of a potential function implies that the state graph is acyclic, which establishes the existence of a Nash equilibrium. The existence of a potential function also implies that the Nash dynamics will converge if one player takes the best response action atomically. We restrict our convergence analysis to this atomic and best-response behavior of players. A game that has a potential function is called a potential game.

First, we note that the Makespan policy corresponds to a potential game. This fact has been observed in various places. In particular, Even-Dar et al. [15] gave several bounds on the speed of convergence to Nash equilibria for this policy. The Randomized policy is the same as the Makespan policy for $B|\ |C_{\max}$ and $P|\ |C_{\max}$. Thus, the Randomized policy also corresponds to a potential game for $B|\ |C_{\max}$ and $P|\ |C_{\max}$. We do not know if the Randomized policy for $R|\ |C_{\max}$ leads to a potential game or not. If it did, this would imply, among other things, that Nash equilibria exist. In the following subsections, we study the convergence for the ShortestFirst and LongestFirst policies.

4.1 Convergence for the ShortestFirst *Policy.*

In Section 2.1, we have shown that Nash equilibria exist for the ShortestFirst policy for $R|\ |C_{\max}$ and can be found in polynomial time. In the following, we show that this game is a potential game and players will converge to a Nash equilibrium. Note that this gives an alternative proof of the existence of Nash equilibria.

Theorem 9 *The* ShortestFirst *policy for* $R|\ |C_{\max}$ *is a potential game.*

PROOF. For any state u, let $c(u)$ be the vector of job completion times sorted in non-decreasing order. We show that as a job switches from one machine to another machine to decrease its completion time, it decreases the corresponding vector c lexicographically. To address potential ties, we assume that the potential function is defined with respect to processing times $p^{\epsilon}_{ij} = p_{ij} + \epsilon^{n-i}$, for some sufficiently small $\epsilon > 0$. In other words, if $p_{ij} = p_{i'j}$ and job i is scheduled before job i' on machine j because $i < i'$, then the processing time p^{ϵ}_{ij} of job i on machine j used for the definition of the potential function c is actually smaller than that of i'.

Suppose the system is in state u and $c(u) = (c_1, c_2, \ldots, c_n)$. Suppose job i with completion time c_i switches machines and decreases its completion time. Call the new state v, and let $c(v) = (c'_1, c'_2, \ldots, c'_n)$. Let i's completion time in v be c'_j. We know that $c'_j < c_i$. However, the change in i's action may cause an increase in the completion times of other jobs. Assume that job i switched to machine k in state v. Jobs whose completion time increases after this switch are the jobs that are scheduled on machine k and whose processing time on machine k is greater than i's processing time on machine k. Thus, the completion times of these jobs in state u (before i moves) were greater than c'_j. Thus in the resulting vector $c(v)$, we decrease an element of the vector from c_i to c'_j and we do not increase any element with value less than c'_j. Thus this switch decreases the corresponding vector lexicographically, i.e., $c(v) < c(u)$, and so c is a potential function. □

Corollary 10 *Selfish behavior of players will converge to a Nash equilibrium under the* ShortestFirst *policy for* $R|\ |C_{\max}$.

Knowing that the selfish behavior of players converges to a Nash equilibrium and the social value of a Nash equilibrium is bounded does not indicate a fast convergence to good solutions. We are interested in the speed of convergence to a Nash equilibrium. We consider the best responses of jobs and prove fast convergence to Nash equilibria for the ShortestFirst policy.

Theorem 11 *For* $R|\ |C_{\max}$ *with the* ShortestFirst *policy, best responses of jobs converge to a Nash equilibrium after* n *rounds in which jobs may make one selfish move each, in arbitrary order. In other words, from any state in the state graph, it takes at most* n^2 *state traversals to end up in a state with no outgoing edges.*

PROOF. In round t, let i_t be the alphabetically first job which achieves the minimum possible disutility among the set of jobs $J_t = \{1, 2, \ldots, n\} - \{i_1, \ldots, i_{t-1}\}$, given the (fixed) strategies of $i_1, \ldots, i_{t-1}$. We prove by induction that in round t, job i_t moves to some machine and remains there in subsequent rounds.

Suppose j is a machine on which i_t achieves its minimum disutility (given the strategies of $i_1, \ldots, i_{t-1}$). Then in round t, a best response for i_t is to move to machine j. We show that this machine is the weakly best response of i_t for *any* set of strategies of jobs J_t. By weakly best response, we mean there is no other action that gives the player a strictly better payoff (i.e., a smaller completion time in our setting).

First notice that the disutility of i_t on j cannot increase as jobs in J_t alter their strategies. This is because any job $i' \in J_t$ has processing time at least

$p_{i_t j}$ on machine j and, upon equality, is alphabetically larger or else we would have set $i_t = i'$ in round t. Now consider some other machine j'. Let c_j be the completion time of job i_t on machine j. Then any job with completion time less than c_j on machine j' in round t must be in $\{i_1, \ldots, i_{t-1}\}$ or else we would have picked this job to be i_t in round t. Thus, the strategies of these jobs are fixed. Let i' be the job on machine j' in round t with the smallest completion time that is at least c_j. If $p_{i_t j'} < p_{i' j'}$, then the strategy of i' and all other jobs scheduled after i' on j' in round t does not affect i_t's disutility for machine j'. If $p_{i_t j'} \geq p_{i' j'}$, then even if i' leaves j' in a subsequent round, the completion time of i_t on j' is still at least i''s completion time on j' in round t, or at least c_j. Thus, it is a weakly best response for i_t to remain on machine j.

This shows that it is a weakly best response for i_t to remain on machine j in all subsequent rounds. □

The next theorem proves that the bound of Theorem 11 is tight.

Theorem 12 *There are instances of* $Q|\ |C_{\max}$ *under the* ShortestFirst *policy for which it takes* n *rounds of best responses of players to converge to a Nash equilibrium.*

PROOF. Let the processing time of job i be i, for $i = 1, 2, \ldots, n$, and let the speed of machine j be $1 + (j-1)\epsilon$, for some sufficiently small $\epsilon > 0$ and $j = 1, 2, \ldots, n$. Suppose job i is on machine i initially, for $i = 1, 2, \ldots, n$. In the first round, if jobs play in the order $n, n-1, \ldots, 1$, each job will go to the fastest machine, i.e., machine n. In the second round, played in the same order, all jobs except job 1 go to machine $n-1$. In round t, jobs $n, n-1, \ldots, t$ move from machine $n-t+2$ to machine $n-t+1$. In the end, job i is scheduled on machine $n-i+1$, and it takes n rounds to converge to this equilibrium. □

4.2 Convergence for the LongestFirst *Policy.*

For the LongestFirst policy, it is possible to prove convergence in the $Q|\ |C_{\max}$, $B|\ |C_{\max}$, and $P|\ |C_{\max}$ models in a manner similar to that of Theorem 11. One just must argue that in each round the job with the longest feasible processing time (among jobs not yet considered) moves to its optimal machine and remains there in subsequent rounds.

Theorem 13 *For the* LongestFirst *policy and* $Q|\ |C_{\max}$, $B|\ |C_{\max}$, *or* $P|\ |C_{\max}$, *the best responses of jobs converge to a Nash equilibrium after* n *rounds in which jobs may make one selfish move each, in arbitrary order.*

We note that Theorem 13 proves the existence of a Nash equilibrium in these games. However, we do not know how to prove that the LongestFirst policy converges in the $R|\ |C_{\max}$ model.

5 Conclusion and Future Work

We have studied abstract scheduling games where the disutility of each player is its (expected) completion time. We note that our results can be applied in many practical network settings. Our results can be directly applied to the Internet setting [33] where there is a set of selfish clients, each of whom must choose a server from a set of servers. Each client tries to minimize its latency, or job completion time; the social welfare is the system-wide latency. Similar problems arise in the context of wireless networks. For example, the basic fairness and load balancing problem in wireless LANs [7] is reduced to the problem of unrelated parallel machine scheduling. Centralized algorithms have been designed for this problem in [7]. We hope to use ideas from our coordination mechanisms to design decentralized algorithms for this problem. In the third generation wireless data networks, the channel quality of a client is typically time-varying [8]; it would be interesting to study coordination mechanisms in this context given that users may exhibit selfish behavior.

Theoretically, the most interesting open problem in this paper is to find coordination mechanisms with constant price of anarchy for $B|\ |C_{\max}$ and $R|\ |C_{\max}$. We have shown that this cannot be achieved by any deterministic coordination mechanism which uses a common tie-breaking rule, and a subsequent result by [3] has ruled out any deterministic coordination mechanism. Another problem left open in this paper is the existence of pure Nash equilibria for the Randomized policy for $R|\ |C_{\max}$.

Throughout this paper, we assumed that all information regarding job processing times was public knowledge. A new direction considered in [10] and [28] is the design of coordination mechanisms in a private information setting, i.e., where a job's processing time is a private value. In such a setting, it would be nice if a coordination mechanism gave incentives for jobs to announce their true processing times. The only constant-factor price of anarchy for $Q|\ |C_{\max}$ and the best factor for $P|\ |C_{\max}$ in our paper are achieved using the LongestFirst policy, but this policy is not truthful. In particular, jobs can artificially inflate their length (e.g., by inserting empty cycles) and as a result actually decrease their disutility. A truthful coordination mechanism with a constant price of anarchy in these settings would be an interesting result.

Acknowledgments. The work of the fourth author was supported by NSF award 0426686.

References

[1] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM*, 44:486–504, 1997.

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

[3] Y. Azar, K. Jain, and V. S. Mirrokni. (Almost) optimal coordination mechanisms for unrelated machine scheduling. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 323–332, 2008.

[4] Y. Azar, J. Naor, and R. Rom. The competitiveness of on-line assignments. *Journal of Algorithms*, 18:221–237, 1995.

[5] A. Bagchi. *Stackelberg Differential Games in Economic Models*, volume 64 of *Lecture Notes in Control and Information Sciences*. Springer, 2004.

[6] M. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation*. Yale University Press, 1956.

[7] Y. Bejerano, S.-J. Han, and L. Li. Fairness and load balancing in wireless LANs using association control. In *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, pages 315–329, 2004.

[8] S. Borst. User-level performance of channel-aware scheduling algorithms in wireless data networks. *IEEE/ACM Transaction on Networking*, 13:636–647, 2005.

[9] Y. Cho and S. Sahni. Bounds for list schedules on uniform processors. *SIAM Journal on Computing*, 9:91–103, 1980.

[10] G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination mechanisms. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 345–357. Springer, 2004.

[11] R. Cole, Y. Dodis, and T. Roughgarden. How much can taxes help selfish routing? In *Proceedings of the 4th ACM Conference on Electronic Commerce*, pages 98–107, 2003.

[12] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 413–420, 2002.

[13] E. Davis and J. M. Jaffe. Algorithms for scheduling tasks on unrelated processors. *Journal of the ACM*, 28:721–736, 1981.

[14] G. Dobson. Scheduling independent tasks on uniform processors. *SIAM Journal on Computing*, 13:705–716, 1984.

[15] E. Even-Dar, A. Kesselman, and Y. Mansour. Convergence time to Nash equilibria. In *Proceedings of the 30th International Colloquium on Automata, Languages and Programming*, volume 2719 of *Lecture Notes in Computer Science*, pages 502–513. Springer, 2003.

[16] G. Finn and E. Horowitz. A linear time approximation algorithm for multiprocessor scheduling. *BIT*, 19:312–320, 1979.

[17] L. Fleischer, K. Jain, and M. Mahdian. Tolls for heterogeneous selfish users in multicommodity networks and generalized congestion games. In *Proceedings of the 45th Symposium on Foundations of Computer Science*, pages 277–285, 2004.

[18] D. K. Friesen. Tighter bounds for LPT scheduling on uniform processors. *SIAM Journal on Computing*, 16:554–560, 1987.

[19] M. Gairing, T. Lücking, M. Mavronicolas, and B. Monien. Computing Nash equilibria for scheduling on restricted parallel links. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 613–622, 2004.

[20] D. K. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang. Optimizing cost and performance for multihoming. *ACM SIGCOMM Computer Communication Review*, 34:79–92, 2004.

[21] R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[22] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17:416–429, 1969.

[23] O. H. Ibarra and C. E. Kim. Heuristic algorithms for scheduling independent tasks on nonidentical processors. *Journal of the ACM*, 24:280–289, 1977.

[24] N. Immorlica, L. Li, V. S. Mirrokni, and A. S. Schulz. Coordination mechanisms for selfish scheduling. In *Proceedings of the 1st International Workshop on Internet and Network Economics*, volume 3828 of *Lecture Notes in Computer Science*, pages 55–69. Springer, 2005.

[25] Y. A. Korilis, A. A. Lazar, and A. Orda. Achieving network optima using Stackelberg routing strategies. *IEEE/ACM Transactions on Networking*, 5:161–173, 1997.

[26] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, volume 1563 of *Lecture Notes in Computer Science*, pages 404–413. Springer, 1999.

[27] J. K. Lenstra, D. B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46:259–271, 1990.

[28] H. Moulin. On scheduling fees to prevent merging, splitting and transferring of jobs. *Mathematics of Operations Research*, 32:266–283, 2007.

[29] J. F. Nash. Equilibrium points in N-person games. *Proceedings of the National Academy of Sciences of the United States of America*, 36:48–49, 1950.

[30] N. Nisan and A. Ronen. Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 129–140, 1999.

[31] T. Roughgarden. Stackelberg scheduling strategies. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 104–113, 2001.

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

[33] S. Suri, C. D. Tóth, and Y. Zhou. Selfish load balancing and atomic congestion games. In *Proceedings of the 16th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 188–195, 2004.

[34] H. von Stackelberg. *Marktform und Gleichgewicht*. Springer, 1934.

[35] T. Vredeveld. *Combinatorial Approximation Algorithms: Guaranteed Versus Experimental Performance*. PhD thesis, Technische Universiteit Eindhoven, The Netherlands, 2002.