Coordination Mechanisms for Weighted Sum of Completion Times*

NYU CIMS Technical Report: TR2010-930

RICHARD COLE †   VASILIS GKATZELIS ‡   VAHAB MIRROKNI §

Abstract

We study policies aiming to minimize the weighted sum of completion times of jobs in the context of coordination mechanisms for selfish scheduling problems. Our goal is to design local policies that achieve a good price of anarchy in the resulting equilibria for unrelated machine scheduling. In short, we present the first constant-factor-approximate coordination mechanisms for this model.

First, we present a generalization of the ShortestFirst policy for weighted jobs, called SmithRule; we prove that it achieves an approximation ratio of 4 and we show that any set of non-preemptive ordering policies can result in equilibria with approximation ratio at least 3 even for unweighted jobs. Then, we present ProportionalSharing, a preemptive strongly local policy that beats this lower bound of 3; we show that this policy achieves an approximation ratio of 2.61 for the weighted sum of completion times and that the EqualSharing policy achieves an approximation ratio of 2.5 for the (unweighted) sum of completion times. Furthermore, we show that ProportionalSharing induces potential games (in which best-response dynamics converge to pure Nash equilibria).

All of our upper bounds are for the robust price of anarchy, defined by Roughgarden [36], so they naturally extend to mixed Nash equilibria, correlated equilibria, and regret minimization dynamics. Finally, we prove that our price of anarchy bound for ProportionalSharing can be used to design a new combinatorial constant-factor approximation algorithm minimizing weighted completion time for unrelated machine scheduling.

---

*This work was supported in part by NSF grant CCF0830516 and by Andreas Mentzelopoulos Scholarships for the University of Patras.

†cole@cims.nyu.edu. Courant Institute, New York University, N.Y.

‡gkatz@cims.nyu.edu. Courant Institute, New York University, N.Y.

§mirrokni@google.com. Google Research, New York, N.Y.

1 Introduction

Traditionally, work in operations research has focused upon finding a globally optimal solution for optimization problems. Computer scientists have also long studied the effects of a lack of different kinds of resources, mainly the lack of computational resources in optimization. Recently, the *lack of coordination*, inherent in many settings, has become an important consideration in designing distributed systems. To address this lack of coordination, decentralized algorithms are being developed for self-interested users. In these algorithms, a central authority can only design protocols and specify rewards hoping that the independent and selfish choices of the users, given the rules and rewards of the protocols, may result in a socially desirable outcome. Also, in order to measure the performance of these algorithms, the global objective function is evaluated at equilibrium points for selfish users. For example, the quality of an algorithm or mechanism can be measured by its price of anarchy [31], which is the worst case ratio of the social cost of a Nash equilibrium over that of a central global social optimum.

In order to achieve a good price of anarchy, several approaches have been proposed, imposing economic incentives on self-interested agents. For example, these incentives may be provided by using monetary payments [6, 14, 21], by enforcing strategies upon a fraction of users, or with the Stackelberg strategy [5, 30, 35, 44]. The main disadvantage of these methods is the need for global knowledge of the system and thus for high communication complexity. In many settings, it is important to be able to compute mechanisms locally. A different approach, which is the focus of our paper, uses *coordination mechanisms*, introduced by Christodoulou, Koutsoupias and Nanavati [13]. Given a set of facilities, a coordination mechanism is a set of *local* policies, one for each facility, that assign a cost to each agent using the facility. The cost assigned to each agent is a function only of the agents who have chosen to use the corresponding facility.

Consider, for example, the *selfish scheduling game* in which there are n jobs owned by independent users, m machines, a processing time p_{ij} for job i on machine j, and a weight (or importance or impatience) w_i for each job i. For now, let us concentrate on the *pure strategies* case where each user selects one machine to assign its job to. Each user has full information about the game and behaves selfishly. Specifically, it wishes to minimize its weighted completion time by assigning its job to the machine on which it yields the earliest completion time. The global objective however, is to minimize either the average completion time or the weighted average completion time of all jobs. A coordination mechanism [13, 29, 3] for this game is a set of local policies, one for each machine, that determines how to schedule jobs assigned to that machine. 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 and local fashion.

In this paper, we mainly study *strongly local* policies in which the policy of each machine j is a function only of the processing time p_{ij} of every job i assigned to machine j. We first consider strongly local *ordering* policies, i.e. deterministic non-preemptive strongly local policies that satisfy the independence of irrelevant alternatives or IIA property, which we define in Section 2. Two examples of such policies are the ShortestFirst and LongestFirst policies in which jobs are ordered in non-decreasing and non-increasing order of their processing times respectively. Later in the paper we also study preemptive strongly local policies like EqualSharing and ProportionalSharing.

Several local policies have been studied for machine scheduling problems, both in the context of greedy or local search algorithms for machine scheduling [28, 20, 37, 17, 1, 4, 8, 45], and also in the context of coordination mechanisms [31, 16, 13, 29, 3, 10, 18]. Previous papers mainly considered the makespan, i.e. the maximum completion time over all jobs, as the social cost function, but, in this paper, we study the weighted sum of completion times instead.

Scheduling problems have long been studied from a centralized optimization perspective. It has proven to be quite convenient to use a standard classification notation due to [24] in order to refer to different variants. Each variant can be denoted by $\alpha|\beta|\gamma$. The first parameter (α) defines the machine model and the last parameter (γ) specifies the objective function to be minimized. The second parameter is used to indicate additional characteristics of the jobs, but for the purpose of this paper it will be left blank.

The problem of minimizing the weighted sum of completion times is NP-complete even for identical machines ($P| \mid \sum w_i c_i$) [32]. For this setting, there exists a PTAS [42], but if we allow for unrelated machines, where p_{ij} values can be arbitrary ($R| \mid \sum w_i c_i$), the problem becomes APX-hard [26]. For the latter model, there exist constant-factor approximation algorithms which are based on rounding optimal solutions of mathematical programming relaxations [25, 38, 40, 41]; our work contributes a new combinatorial constant-factor approximation algorithm for this setting. On the other hand, minimizing the (unweighted) sum of completion times is polynomial time solvable even for unrelated machines ($R| \mid \sum c_i$) using matching techniques [27, 9]. For identical machines ($P| \mid \sum c_i$), the ShortestFirst policy, leads to an optimal schedule at any pure Nash equilibrium point[1] [15]. For a good survey of results regarding the average (weighted) completion time objective function, see [33, Chapter 11].

[1]In [29] it is shown that these equilibria are exactly the solutions generated by the shortest-first greedy algorithm.

Our Results. We study coordination mechanisms aiming to minimize the weighted sum of completion times for the unrelated machine scheduling problem. In particular, we present the first constant-factor approximate mechanisms for this problem, and we show an intrinstic advantage of preemptive policies over non-preemptive ones in this context. Our results also imply a new combinatorial constant-factor approximation algorithm for $R| \mid \sum w_i c_i$.

First, we study a generalization of the ShortestFirst policy, called SmithRule. For each machine j and any set of jobs assigned to it, this policy follows Smith's rule [43], that is, it orders the jobs giving higher priority to a job i with smaller $\frac{p_{ij}}{w_i}$ value. We prove that the price of anarchy of SmithRule is at most 4. Furthermore, we show that any set of non-preemptive strongly local policies with the IAA property (a.k.a. ordering policies) may result in pure Nash equilibria with approximation ratio at least 3 even for unweighted jobs. This gives a lower bound of 3 for the pure price of anarchy of any set of non-preemptive ordering policies.

Next, we present a preemptive strongly local policy that beats this lower bound of 3. In particular, we study the EqualSharing policy [18] that gives an equal share of processing time to each active job at each time, and a generalization of this policy, called ProportionalSharing, that gives each job a share of the processing time proportional to the ratio of its weight over the sum of the weights of all jobs being processed on the same machine at each time. We prove an approximation ratio of 2.5 for the average completion time for the EqualSharing policy and an approximation ratio of 2.61 for the weighted average completion time for the ProportionalSharing policy, beating the non-preemptive ordering policies' lower bound of 3. This is in contrast to the makespan social function in which the EqualSharing policy achieves an approximation ratio of $\Theta(m)$ [18], which is no better than other non-preemptive policies such as ShortestFirst. Furthermore, we show that ProportionalSharing results in potential games by presenting an exact potential function. This proof, in turn, shows that best-response dynamics of players converge to pure Nash equilibria.

All of our upper bounds are for the *robust price of anarchy*, defined by Roughgarden [36] (see Section 2) and thus, our bounds also hold for mixed Nash equilibria, correlated equilibria, and regret minimization dynamics. Finally, we note that our price of anarchy bound for ProportionalSharing yields a new combinatorial constant-factor approximation algorithm for minimizing the weighted sum of completion times for the unrelated machine scheduling problem. To the best of our knowledge, all the previous constant-factor approximation algorithms for this problem were based on rounding optimal solutions of mathematical programming relaxations [25, 38, 40, 41] so the algorithm that we present is the first combinatorial one to achieve a constant-factor approximation.

Other Related work. Coordination mechanism design was introduced in [13] by Christodoulou, Koutsoupias and Nanavati. In their paper, they analyzed the LongestFirst policy w.r.t. the makespan for identical machines ($P| |C_{\max}$) and also studied a selfish routing game. Immorlica, Li, Mirrokni, and Schulz [29] study four coordination mechanisms for four types of machine scheduling problems and survey the results for these problems. They further study the speed of convergence to equilibria and the existence of pure Nash equilibria for the ShortestFirst and LongestFirst policies. Azar, Jain, and Mirrokni [3] showed that the ShortestFirst policy and any set of non-preemptive strongly local policies with the IAA property do not achieve an approximation ratio better than $\Omega(m)$. Additionally, they presented a non-preemptive local policy that achieves an approximation ratio of $O(\log m)$ and a policy that induces a potential game and gives an approximatition ratio of $O(\log^2 m)$. Caragiannis [10] showed an alternative $O(\log m)$-approximate coordination mechanism that minimizes makespan for unrelated machine scheduling and does lead to a potential game. Fleischer and Svitkina [22] show a lower bound of $\Omega(\log m)$ for all non-preemptive local policies with the IAA property.

More recently, Dürr and Thang proved that the EqualSharing policy results in a potential game, and achieves a price of anarchy of $\Theta(m)$ for $R| |C_{\max}$. In the context of coordination mechanisms, an instance for which preemptive policies have an advantage over non-preemptive ones was also shown by Caragiannis [10]; he presented a local preemptive policy of $O(\log m/\log\log m)$ beating the lower bound of $\Omega(\log m)$ that Fleischer and Svitkina [22] show for all local non-preemptive mechanisms. However, his preemptive policy, unlike ProportionalSharing, doesn't induce a potential game which would guarantee the existence of pure Nash equilibria.

Coordination mechanisms are related to local search algorithms. Starting from a solution, a local search algorithm iteratively moves to a neighbor solution which improves the global objective. This is based on a neighborhood relation that is defined on the set of solutions. The local improvement moves in the local search algorithm correspond to the best-response moves of users in the game defined by the coordination mechanism. The speed of convergence and the approximation factor of local search algorithms for scheduling problems have been studied in several papers [17, 19, 20, 28, 37, 39, 45, 1, 4]. Vredeveld surveyed some of the results on local search algorithms for scheduling problems in his thesis [45].

Finally, another widely studied scheduling policy is the Makespan policy in which all jobs on the same machine are processed in parallel so that the completion time of every job on machine j is the makespan of machine j. The price of anarchy of this policy is unbounded even for two machines. Tight price of anarchy results for (mixed) Nash equilibria are known for this policy for special cases of the unrelated scheduling problem [16, 2, 23, 31].

2 Preliminaries

Throughout this paper, let N be a set of n jobs to be scheduled on a set M of m machines. Each job needs to be assigned to exactly one machine and each machine can process only one job at any time. In selfish scheduling problems, each job is owned by a selfish agent who decides which machine it will be scheduled on; see below for details. Let p_{ij} denote the processing time of job $i \in N$ on machine $j \in M$ and let w_i denote its weight (or importance or impatience). Our goal is to minimize the weighted sum of the completion times of the jobs, i.e. $\sum_{i=1}^{n} w_i c_i$, where c_i is the completion time of job i.

The main scheduling model we study is *unrelated* machine scheduling (i.e., $R| \mid \sum w_i c_i$) in which p_{ij}'s are arbitrary. Another machine scheduling model we consider is the *restricted identical* machines model ($B| \mid \sum w_i c_i$), in which each job i can be scheduled only on a subset T_i of the machines, i.e., $p_{ij} = p_i$ if $j \in T_i$ and $p_{ij} = \infty$ otherwise. A third model is that of *restricted related* machines in which each machine j has a speed q_j and each job i has a processing requirement p_i. A job i can be scheduled only on a subset T_i of the machines, with processing time $p_{ij} = p_i/q_j$ if $j \in T_i$, and $p_{ij} = \infty$ otherwise. The restricted identical machines model is a special case of the restricted related machines model.

A *coordination mechanism* is a set of local policies, one for each machine, that determines how to schedule the jobs assigned to that machine. It thereby defines a game in which there are n agents (jobs) and each agent's strategy set is the set of machines M. Given a strategy profile s, the disutility of job i is its weighted completion time $w_i c_i(s)$ as defined by the coordination mechanism. The goal of each job is to choose a strategy (i.e., a machine) that minimizes its disutility.

We consider a *normal-form game* among selfish jobs resulting from a coordination mechanism, and study its equilibria. The social cost function with respect to which we will be measuring the inefficiency of different schedules is the weighted sum of the completion times, i.e. $\sum_{i=1}^{n} w_i c_i(s)$, where $c_i(s)$ is the completion time of job i in configuration s. We will also be considering the unweighted sum of completion times. The goal of our coordination mechanisms is therefore the creation of the right incentives for the players, such that selfish behavior leads to equilibrium points with low social cost values.

Given a normal-form game, a strategy profile (or a vector of strategies) s is a *Nash equilibrium* if no player has an incentive to change its strategy, i.e. the machine which it is being proccesed by. A game is a *potential game* if there exists a lower bounded potential function over strategy profiles such that any player's deviation leads to a drop of the potential function if and only if its cost drops. A potential game is an *exact* potential game if after each move, the difference of the potential function is equal to that of the player's cost. It is easy to see that a potential game always possesses a pure Nash equilibrium.

The coordination mechanisms we study in this paper use the same local policy on each machine, so we may henceforth refer to a coordination mechanism using the name of the policy. We analyze one non-preemptive strongly local policy that follows Smith's rule [43] and one preemptive strongly local policy, which we call ProportionalSharing. The SmithRule policy is a generalization of the ShortestFirst policy and is defined as follows: given a set of jobs that are assigned to machine j, it orders the jobs giving higher priority to a job i with smaller $\frac{p_{ij}}{w_i}$ value. In case of a tie, the job with the shorter processing time p_{ij} gets higher priority and if both the ratio and the processing times tie, a global tie breaking rule is used. The ProportionalSharing policy is a generalization of the EqualSharing policy and it schedules the jobs in parallel using time-multiplexing giving each job a share of the processor time proportional to the ratio of its weight over the sum of the weights of all jobs being processed on the same machine. It is easy to see that if we assume unit weights, these policies turn into ShortestFirst and EqualSharing respectively. As is already known [43], given a strategy profile, the social cost is minimized if we order the jobs within each machine using SmithRule. For any configuration s, we will let $w_i c_i^{\alpha}(s)$ and $C^{a}(s)$ denote the cost for player i and the social cost respectively, where $\alpha \in \{SR, PS, SF, ES\}$ denotes the policy, namely SmithRule, ProportionalSharing, ShortestFirst and EqualSharing, respectively. Finally, slightly abusing notation, let $S_j = \{i \in N \mid s_i = j\}$ denote the set of jobs that have chosen machine j in configuration s.

A local policy for machine j uses only the information about the jobs on the same machine j, but it can look at all the parameters of these jobs, including their processing times on other machines. On the other hand, a *strongly local* policy may only depend on the processing time that these jobs have on this machine j. We say that a policy satisfies the *independence of irrelevant alternatives* or *IIA* property if for any set S of jobs and any two jobs $i, i' \in S$, if i has a smaller completion time than i' in S, then i should have a smaller completion time than i' in any set $S \cup \{k\}$. In other words, whether i or i' is preferred should not be changed by the availability of job k. The IIA property appears as an axiom in voting theory, bargaining theory and logic.

In order to quantify the inefficiency caused by the lack of coordination, we use the notion of *Price of Anarchy* (PoA) introduced by Koutsoupias and Papadimitriou [31], that is, the ratio between the social cost value of the worst Nash equilibrium and that of the optimum schedule. More specifically, we prove upper bounds for the *robust PoA*, recently introduced by Roughgarden [36], which imply the same bounds for the inefficiency of pure and mixed Nash equilibria, correlated equilibria, and for regret minimization dynamics [7]. More specifically, Roughgarden defines a cost-minimization game to be (λ, μ)-smooth if for every two outcomes s and s^*,

$$\sum_{i \in N} c_i(s_{-i}, s_i^*) \leq \lambda \sum_{i \in N} c_i(s^*) + \mu \sum_{i \in N} c_i(s).$$

The *robust price of anarchy* of such a game is then equal to:

$$\inf \left\{ \frac{\lambda}{1-\mu} : (\lambda, \mu) \text{ s.t. the game is } (\lambda, \mu)\text{-smooth} \right\}.$$

To be more precise, we are interested in upper bounds for the PoA of coordination mechanisms rather than the PoA of specific games. Based on [13], the price of anarchy of a coordination mechanism is the maximum ratio over all the games G that the mechanism may induce, of the social cost of a Nash equilibrium of G, divided by the optimum social cost of the scheduling problem underlying G. It is important to note that this optimum social cost depends only on the p_{ij}'s and not on the coordination mechanism. Given this definition, we define a coordination mechanism α to be (λ, μ)-smooth if for every two outcomes s and s^* of any game that it may induce,

$$\sum_{i \in N} c_i^{\alpha}(s_{-i}, s_i^*) \leq \lambda \sum_{i \in N} c_i(s^*) + \mu \sum_{i \in N} c_i^{\alpha}(s).$$

In our context, we know that the optimal schedule uses SmithRule for weighted jobs and EqualSharing for unweighted jobs, so we will be using $c_i^{SR}(s^*)$ or $c_i^{ES}(s^*)$ respectively for the optimum social costs. We can now define the robust price of anarchy of a coordination mechanism to be equal to:

$$\inf \left\{ \frac{\lambda}{1-\mu} : (\lambda, \mu) \text{ s.t. the coordination mechanism is } (\lambda, \mu)\text{-smooth} \right\}.$$

In [36], a class of games is defined to be *tight* if there exists a game instance with pure PoA equal to the upper bound of the robust PoA of all games in this class[2], which is the case for many of the classes of games that we study in this paper.

3 Non-Preemptive Coordination Mechanisms

3.1 SmithRule policy

It is known that in order to minimize the weighted sum of completion times on one machine, SmithRule is optimum [43]. Here, we show that using this rule in

[2] Or more generally, the upper bound equals the supremum of the pure PoA over those game instances for which a pure Nash equilibrium exists.

our game-theoretic model for multiple machines will result in Nash equilibria with social cost a constant-factor away from the optimum.

Theorem 3.1. *The robust PoA of* SmithRule *for* $R| \mid \sum w_i c_i$ *is at most* 4.

Proof. We start by giving a rough intuition. Assume that all jobs start from the optimum configuration s^* and we give each job i credit equal to four times its cost in s^*, i.e. $4w_i c_i^{SR}(s^*)$. Now, given any pure Nash equilibrium s, each job i moves from machine s_i^* to machine s_i; in doing so, it "gathers" credit of value at least twice its cost in s (i.e. $2w_i c_i^{SR}(s)$), and then gives half of it away to some of the jobs who used to be in its current machine s_i, "redeeming itself" for pushing their position on s_i higher. Job i "gathers" its credit from those jobs which, in turn, moved into i's initial machine s_i^* thus increasing i's cost for uniquely deviating back (from s_i to s_i^*). The remaining credit of each job will be at least as much as its cost in equilibrium s and the conclusion is that four times the social cost value at s^* is greater or equal to the social cost value at any equilibrium s, thus proving the bound.

In order to prove the above, it would suffice if we could show that for every machine j, the incoming credit of the jobs in S_j, along with the initial credit of the jobs in S_j^*, is enough for every job $i \in S_j^*$ to have twice its cost for uniquely deviating back to $j = s_i^*$, or

$$2 \sum_{i \in S_j^*} w_i c_i^{SR}(s_{-i}, s_i^*) \leq \sum_{i \in S_j} w_i c_i^{SR}(s) + 4 \sum_{i \in S_j^*} w_i c_i^{SR}(s^*).$$

In order to prove a bound on the more general notion of the robust PoA, we instead let s and s^* be *any* two configurations; then we want to show that for every machine j,

$$\sum_{i \in S_j^*} w_i c_i^{SR}(s_{-i}, s_i^*) \leq \frac{1}{2} \sum_{i \in S_j} w_i c_i^{SR}(s) + 2 \sum_{i \in S_j^*} w_i c_i^{SR}(s^*).$$

In order to show this, we first prove that this inequality only becomes tighter if for any two jobs $i, i' \in S_j \cup S_j^*$, their ratios on j are equal, i.e. $\frac{p_{ij}}{w_i} = \frac{p_{i'j}}{w_{i'}}$. Assume that not all ratios are equal and let $Max_j = \{i \in S_j \cup S_j^* \mid \forall i' \in S_j \cup S_j^*, \frac{p_{ij}}{w_i} \geq \frac{p_{i'j}}{w_{i'}}\}$ be the set of jobs of maximum ratio among the two sets. Also, let $J^* = Max_j \cap S_j^*$ and $J = Max_j \cap S_j$ be the maximum ratio jobs in sets S_j^* and S_j respectively.

For all the jobs $i \in Max_j$, we decrease their ratio $\frac{p_{ij}}{w_i}$ by the minimum positive value Δ such that the cardinality of Max_j increases. In order to do this, we decrease the processing time of each job $i \in Max_j$ by $w_i \Delta$ and then we reorder the jobs so that they obey the SmithRule policy (so that jobs with equal ratio are ordered in a ShortestFirst fashion). After a change of this sort, in order to show that the inequality may only become tighter, we want the drop of the LHS to be less than or equal to the drop of the RHS, so it suffices to show that:

$$\sum_{i \in J^*} w_i \left(w_i + \sum_{i' \in J} w_{i'} \right) \Delta \leq \frac{1}{2} \sum_{i \in J} w_i \left(\sum_{(i' \in J) \wedge (w_{i'} \leq w_i)} w_{i'} \right) \Delta + 2 \sum_{i \in J^*} w_i \left(\sum_{(i' \in J^*) \wedge (w_{i'} \leq w_i)} w_{i'} \right) \Delta.$$

The LHS actually corresponds to an upper bound for this drop, which we get if we assume that every job $i \in J^*$ that deviates back to machine j is processed last. On the other hand, the RHS corresponds to a lower bound for the corresponding drop. To be more specific, this drop corresponds to the case when no reordering of the jobs takes place after the processing times' modifications. This is indeed a lower bound, since the reordering would only lead to an even greater drop. If we set $A = \sum_{i \in J^*} w_i$ and $B = \sum_{i \in J} w_i$ and further notice that:

$$2 \sum_{i \in J^*} w_i \left(\sum_{(i' \in J^*) \wedge (w_{i'} \leq w_i)} w_{i'} \right) = A^2 + \sum_{i \in J^*} w_i^2,$$

and similarly for jobs in J, the inequality becomes:

$$AB + \sum_{i \in J^*} w_i^2 \leq \frac{1}{4} \left(B^2 + \sum_{i \in J} w_i^2 \right) + A^2 + \sum_{i \in J^*} w_i^2,$$

or equivalently,

$$\left(\frac{B}{2} - A \right)^2 + \frac{1}{4} \sum_{i \in J} w_i^2 \geq 0,$$

which is true for any value of A, B and weights of jobs in J.

We can now assume that for any job $i \in S_j \cup S_j^*$, $\frac{p_{ij}}{w_i} = r$ for some $r \in \mathbb{R}^+$. Since all jobs have the same ratio on j, the sums on the right hand side of the initial inequality are minimized when the jobs are in order of non-decreasing processing time and thus suffices to show that:

$$\sum_{i \in S_j^*} w_i \left(p_{ij} + \sum_{i' \in S_j} p_{i'j} \right) \leq \frac{1}{2} \sum_{i \in S_j} w_i \left(\sum_{(i' \in S_j^*) \wedge (p_{i'j} \leq p_{ij})} p_{i'j} \right) + 2 \sum_{i \in S_j^*} w_i \left(\sum_{(i' \in S_j^*) \wedge (p_{i'j} \leq p_{ij})} p_{i'j} \right).$$

If we replace p_{ij} with $w_i r$ for each job i, we get:

$$rAB + r\sum_{i\in J^*} w_i^2 \leq \frac{1}{4}r\left(B^2 + \sum_{i\in J} w_i^2\right) + rA^2 + r\sum_{i\in J^*} w_i^2,$$

which we already know is true. Now that we have proved that the initial inequality is true for all machines j, summing up the corresponding inequalities over all $j \in M$ gives:

$$\sum_{i\in N} w_i c_i^{SR}(s_{-i}, s_i^*) \leq \frac{1}{2}C^{SR}(s) + 2C^{SR}(s^*).$$

This shows that this coordination mechanism is $(2, 1/2)$-smooth, therefore showing an upper bound of 4 for its robust price of anarchy and proving the theorem.

In order to get the pure PoA bound, we need only consider the case when s is an equilibrium and s^* is an optimum configuration. For all players $i \in N$, we know that $w_i c_i^{SR}(s) \leq w_i c_i^{SR}(s_{-i}, s_i^*)$, thus:

$$C^{SR}(s) \leq \frac{1}{2}C^{SR}(s) + 2C^{SR}(s^*) \Rightarrow \frac{C^{SR}(s)}{C^{SR}(s^*)} \leq 4.$$

□

3.2 Lower Bounds for Non-preemptive Policies

In this section, we study non-preemptive strongly local policies with the IIA property, and show a lower bound for the PoA of any set of such policies.

Theorem 3.2. *The pure price of anarchy for any set of non-preemptive strongly local policies satisfying the IIA property for* $R| \,|\sum w_i c_i$ *is at least* 3*. This lower bound holds even for the* unweighted *variant, i.e.,* $R| \,|\sum c_i$.

Proof. In the proof, we use the notion of a "game graph" from [11] (section 3). In this directed graph, each machine corresponds to a vertex and each job corresponds to an edge; the edge is directed from the machine the job is assigned to in the optimal configuration to the machine the job uses in the Nash equilibrium configuration. All jobs need infinite processing time on all machines except the ones they use in one of these two configurations.

We construct a lower bound game graph G starting from a complete binary tree of height h with edges directed toward the root. Each vertex j is assigned a number p_j that equals the processing time of any job that is assigned to the corresponding machine j in either of the two configurations. The root of the tree is given processing time 1. For every other vertex, except the leaves of the tree, let x be the processing time of its parent; if the vertex is a left child, its processing time is $x/3$, while if it's a right child, its processing time is $2x/3$. For a leaf, its processing time is x for a left child and $2x$ for a right child, where once again x is the processing time of its parent. What remains is to define the assignment of jobs in the optimal configuration (one job per machine). The Nash equilibrium configuration will then follow directly as each job will move to the machine that its outgoing edge points to.

We start with a set A of $(2^{h+1}-2)$ jobs and a vertex j in the h-th level (one level above the leaves). Since j's policy is strongly local, it only looks at the processing time of jobs assigned to itself and their IDs. As a result, if the processing time of all jobs in A on machine j was p_j, then it would be an ordering policy that would order the jobs based on a global ordering of IDs. We assign the job ordered first by such an ordering policy to j's left child and the job ordered second to its right child and remove them from A. By the IIA property we know that, no matter whether the remaining jobs of A will be available to j or not, these two will still be the first and second ones in the actual ordering. We continue in a similar fashion for all machines in the h-th level. Then, we perform the same process for the $(h-1)$-th level and so on, until we reach the root (to which we don't assign any job).

In the optimal configuration that we defined, each machine has one job assigned to it. The sum of completion times of each level except the root and the leaves is 1 and the sum of completion times of all the leaves is 3, thus leading to an optimal configuration cost of $(h-1)+3$. On the other hand, in the Nash equilibrium configuration, on each machine, the left child goes first and the right child goes second and if the parent deviates back to its optimal strategy, it goes third. This leads each level except the leaves to have sum of completion times equal to 3 and thus a Nash equilibrium configuration cost of $3(h-1)$. For values of h going to infinity, this leads to PoA arbitrarily close to 3. □

Theorem 3.3. *The pure price of anarchy for any set of non-preemptive strongly local policies satisfying the IIA property for* restricted identical *machines ($B| \mid \sum w_i c_i$) is at least* 2.182. *This is true even for the unweighted case, i.e., ($B| \mid \sum c_i$).*

Proof. We show that the lower bound is true for the ShortestFirst policy and it is easy to generalize it to any strongly local ordering policy.
Assume there are $m = 4096$ machines. In the optimal configuration s^*, 1 unit job assigned to machines 257 to 4096. Machines 129 to 256 have 2 unit jobs assigned to them, while machines 65 to 128 have 3 unit jobs. Machines 33 to 64 have 4 unit jobs assigned to them and machines 9 to 32 have 5 unit jobs. Machines 5 to 8 have 6 unit jobs, machines 3 and 4 have 7 unit jobs and finally machine 2 has 8 unit jobs and machine 1 has 10. This leads to $C^{SF}(s^*) = 5519$. All the jobs

that are scheduled first on some machine j are restricted to only being assigned to machines j' with $j' \leq j$ while all other jobs are restricted to the one machine that they are assigned to in s^*.

In the equilibrium s, the jobs that are scheduled first on machines j with $j \in (2^{k-1}, 2^k]$ are now assigned to machines $j - 2^{k-1}$ where $k = 1, 2, ..., 12$. All the remaining jobs obviously stay put. This leads to $C^{SF}(s) = 12044$ and therefore $PoA \geq \frac{C^{SF}(s)}{C^{ES}(s^*)} = 2.182$. □

4 Preemptive Coordination Mechanisms

In this section, we study preemptive coordination mechanisms and show that these mechanisms are strictly better w.r.t. the PoA than any set of non-preemptive strongly local policies. These results create a clear dichotomy between all local ordering policies and the EqualSharing and ProportionalSharing preemptive policies w.r.t. the PoA. An intuition for this result is that both EqualSharing for sum of completion times and ProportionalSharing for weighted sum of completion times give high priority players an incentive to avoid crowded machines, although they might have a small processing time there. In contrast, for non-preemptive ordering policies, the cost of high priority players is not affected by the number of other players using the same machine, thus allowing them to move to these machines, possibly resulting in more crowded machines. Another advantage of these coordination mechanisms, is that, unlike SmithRule, they can deal with anonymous jobs, i.e. jobs that don't have IDs.

Before giving the analysis for the ProportionalSharing policy for $\sum w_i c_i$, we first study EqualSharing for the unweighted variant ($\sum c_i$). Before we embark on showing our upper bounds, we begin by proving the following lemma which gives a tighter version of an inequality used by Christodoulou and Koutsoupias [12].

Lemma 4.1. *For every pair of non-negative integers k and k^*,*

$$k^*(k+1) \leq \frac{1}{3}k^2 + \frac{5}{3}\frac{k^*(k^*+1)}{2}.$$

Proof. After some algebra, this translates to showing that for all non-negative integers k and k^*,

$$5k^{*2} + 2k^2 - 6k^*k - k^* \geq 0.$$

We start by taking the partial derivative of the LHS w.r.t. k, i.e. $4k - 6k^*$, from which we infer that for any given value of k^*, the LHS is minimized when $k = \frac{3}{2}k^*$. On substituting this into our inequality, we obtain:

$$5k^{*2} + 2(\frac{3}{2}k^*)^2 - 6k^*\frac{3}{2}k^* - k^* \geq 0 \Rightarrow k^{*2} \geq 2k^*,$$

which is true for $k^* = 0$ and $k^* \geq 2$. For $k^* = 1$ our inequality becomes $k^2 - 3k + 2 \geq 0$ which is true for all non-negative integers k. □

Theorem 4.2. *The robust PoA of* EqualSharing *for* $R| \mid \sum c_i$ *is at most 2.5. This bound is tight even for restricted related machine scheduling.*

Proof. Following an argument very similar with the one for the proof of Theorem 3.1, we prove that for any machine j:

$$\sum_{i \in S_j^*} c_i^{ES}(s_{-i}, s_i^*) \leq \frac{1}{3} \sum_{i \in S_j} c_i^{ES}(s) + \frac{5}{3} \sum_{i \in S_j^*} c_i^{SF}(s^*).$$

In order to show this, we first prove that this inequality only becomes tighter if for any two jobs $i, i' \in S_j \cup S_j^*$, their processing times on j are equal, i.e. $p_{ij} = p_{i'j}$. Assume that not all processing times are equal and let $Max_j = \{i \in S_j \cup S_j^* \mid \forall i' \in S_j \cup S_j^*,\ p_{ij} \geq p_{i'j}\}$ be the set of jobs of maximum processing time among the two sets. Also, let $k^* = |Max_j \cap S_j^*|$ and $k = |Max_j \cap S_j|$ be the number of maximum size jobs in sets S_j^* and S_j respectively.

For all the jobs $i \in Max_j$, we decrease p_{ij} by the minimum positive value Δ such that the cardinality of Max_j increases. After a change of this sort, the LHS drops by $(k^*(k+1))\Delta$ while the RHS drops by $(\frac{1}{3}k^2 + \frac{5}{3}\frac{k^*(k^*+1)}{2})\Delta$. Given Lemma 4.1 above, we conclude that the drop of the LHS is always less than or equal to the drop of the RHS. Using the same inequality again, we conclude that for unit jobs on machine j the inequality is always true; summing up over all $j \in M$ yields:

$$\sum_{i \in N} c_i^{ES}(s_{-i}, s_i^*) \leq \frac{1}{3} \sum_{i \in N} c_i^{ES}(s) + \frac{5}{3} \sum_{i \in N} c_i^{SF}(s^*).$$

This shows that this coordination mechanism is $(5/3, 1/3)$-smooth, therefore showing an upper bound of 2.5 for its robust price of anarchy and proving the theorem.

In order to get the pure PoA bound, we can once again assume that s is an equilibrium and s^* is an optimal configuration w.r.t. the ShortestFirst policy. For all players $i \in N$, we know that $w_i c_i^{ES}(s) \leq w_i c_i^{ES}(s_{-i}, s_i^*)$, thus:

$$\sum_{i \in N} c_i^{ES}(s) \leq \frac{1}{3} \sum_{i \in N} c_i^{ES}(s) + \frac{5}{3} \sum_{i \in N} c_i^{SF}(s^*) \Rightarrow \frac{\sum_{i \in N} c_i^{ES}(s)}{\sum_{i \in N} c_i^{SF}(s^*)} \leq \frac{5}{2}.$$

The tightness of the bound follows from Theorem 3 of [11]. The authors present a load balancing game lower bound, but if we assume that all jobs have unit size and the machines are using EqualSharing, the same proof yields a *pure* PoA lower bound for restricted related machines and unweighted jobs. □

Now we switch to studying the ProportionalSharing policy for the $\sum w_i c_i$ social objective function. We start by proving that the order of completion times of jobs on a machine that follows ProportionalSharing is the same as the order that results from SmithRule; we also present a convenient formula that expresses each job's completion time as a function of itself and the jobs that complete before and after it on the same machine.

Lemma 4.3. *For any machine j using* ProportionalSharing *and any pair of jobs a, b assigned to this machine, $c_a \leq c_b \Leftrightarrow \frac{p_{aj}}{w_a} \leq \frac{p_{bj}}{w_b}$. Also, assuming $c_i \leq c_{i'}$ for any two jobs $i \leq i'$, the completion time of some job b on this machine is:*

$$c_b = p_{bj} + \sum_{a<b} p_{aj} + \sum_{c>b} \frac{p_{bj} w_c}{w_b}.$$

Proof. In order to prove the first part of the lemma, we consider some processing time interval of length L, during which no job is completed and therefore the sum of the weights of the jobs being processed is fixed and equal to W. During this time interval, every player i whose job is being processed gets a share of the processing time equal to $\frac{w_i}{W} L$ and therefore it gets a fraction $\frac{w_i}{p_{ij} W} L$ of its whole job completed. It is easy to notice that jobs with smaller $\frac{p_{ij}}{w_i}$ value complete a greater fraction of their job during any such interval of time and therefore these jobs must be completed first.

To prove the second part of the lemma, we notice that the completion time of job b is only affected by the amount of "work" that the processor has completed by that time and not by the way this processing time has been shared among the jobs. For job b and any job $a < b$, we know that their whole processing demand p_{bj} and p_{aj} respectively has been served. On the other hand, while job b is not complete, for each w_b units of processing time that it receives, any job $c > b$ receives w_c units. Thus, when job b is completed, the processing time spent on any job $c > b$ will be exactly $\frac{p_{bj} w_c}{w_b}$. Adding all these processing times gives the second part of the lemma. □

Before proving the PoA bounds for this policy, we show that this policy also leads to a potential game by generalizing Theorem 3 of [18].

Theorem 4.4. *The* ProportionalSharing *coordination mechanism induces exact potential games.*

Proof. We show that $\Phi(s) = \frac{1}{2} \sum_{i' \in N} w_{i'} \left(c_{i'}(s) + p_{i' s_{i'}} \right)$ is an exact potential function for any such game. Let player i make a better response move from machine x to machine y, decreasing its cost from $w_i c_i(s_{-i}, x)$ to $w_i c_i(s_{-i}, y)$ and let

$$
\begin{aligned}
A &= \left\{a \in N \mid (s_a = x) \wedge \left(\frac{p_{ax}}{w_a} < \frac{p_{ix}}{w_i}\right)\right\}, \\
B &= \left\{b \in N - i \mid (s_b = x) \wedge \left(\frac{p_{bx}}{w_b} \geq \frac{p_{ix}}{w_i}\right)\right\}, \\
C &= \left\{c \in N \mid (s_c = y) \wedge \left(\frac{p_{cy}}{w_c} < \frac{p_{iy}}{w_i}\right)\right\}, \\
D &= \left\{d \in N - i \mid (s_d = y) \wedge \left(\frac{p_{dy}}{w_d} \geq \frac{p_{iy}}{w_i}\right)\right\}.
\end{aligned}
$$

By Lemma 4.3, the drop of the cost of player i after this move is:

$$
\begin{aligned}
w_i c_i(s_{-i}, x) - w_i c_i(s_{-i}, y) = w_i &\left(p_{ix} + \sum_{a \in A} p_{ax} + \sum_{b \in B} \frac{p_{ix} w_b}{w_i}\right) \\
- w_i &\left(p_{iy} + \sum_{c \in C} p_{cy} + \sum_{d \in D} \frac{p_{iy} w_d}{w_i}\right),
\end{aligned}
$$

which is equivalent to:

$$
\begin{aligned}
w_i c_i(s_{-i}, x) - w_i c_i(s_{-i}, y) &= w_i p_{ix} - w_i p_{iy} \\
&+ \sum_{a \in A} w_i p_{ax} + \sum_{b \in B} w_b p_{ix} - \sum_{c \in C} w_i p_{cy} - \sum_{d \in D} w_d p_{iy}.
\end{aligned}
$$

After this deviation of player i, all jobs except i have the same processing time and the only completion times that are affected are those of player i and of the jobs in one of the four sets defined above. More specifically, using Lemma 4.3, we can break down the potential function drop as follows:

$$
\begin{aligned}
2(\Phi(s_{-i}, x) - \Phi(s_{-i}, y)) &= w_i \left(c_i(s_{-i}, x) - c_i(s_{-i}, y) + p_{ix} - p_{iy}\right) \\
&+ \sum_{a \in A} w_a \frac{p_{ax} w_i}{w_a} + \sum_{b \in B} w_b p_{ix} - \sum_{c \in C} w_c \frac{p_{cy} w_i}{w_c} - \sum_{d \in D} w_d p_{iy},
\end{aligned}
$$

which is equivalent to:

$$
\begin{aligned}
2(\Phi(s_{-i}, x) - \Phi(s_{-i}, y)) &= w_i c_i(s_{-i}, x) - w_i c_i(s_{-i}, y) + w_i p_{ix} - w_i p_{iy} \\
&+ \sum_{a \in A} w_i p_{ax} + \sum_{b \in B} w_b p_{ix} - \sum_{c \in C} w_i p_{cy} - \sum_{d \in D} w_d p_{iy},
\end{aligned}
$$

from which one can conclude that:

$$
2(\Phi(s_{-i}, x) - \Phi(s_{-i}, y)) = 2(w_i c_i(s_{-i}, x) - w_i c_i(s_{-i}, y)).
$$

□

We are now ready to prove the PoA bound for ProportionalSharing policy. The intuition behind the following analysis is that the completion time(s) of the last job(s) (the job(s) with the greatest ratio value) is exactly the sum of the processing times of all the jobs that are assigned to the machine.

Theorem 4.5. *The robust PoA of* ProportionalSharing *for* $R|\,|\sum w_i c_i$ *is at most* $\phi+1 = \frac{3+\sqrt{5}}{2} \approx 2.618$. *Moreover, this bound is tight even for the restricted related model.*

Proof. Let s and s^* be any two configurations. We start by showing that for every machine j:

$$\sum_{i \in S_j^*} w_i c_i^{PS}(s_{-i}, s_i^*) \leq \frac{1}{2\phi} \sum_{i \in S_j} w_i c_i^{PS}(s) + \frac{\phi+2}{2} \sum_{i \in S_j^*} c_i^{SR}(s^*).$$

In order to show this, we first prove that this inequality only becomes tighter if for any two jobs $i, i' \in S_j \cup S_j^*$, their ratios on j are equal, i.e. $\frac{p_{ij}}{w_i} = \frac{p_{i'j}}{w_{i'}}$. Assume that not all ratios are equal and let $Max_j = \{i \in S_j \cup S_j^* \mid \forall i' \in S_j \cup S_j^*, \frac{p_{ij}}{w_i} \geq \frac{p_{i'j}}{w_{i'}}\}$ be the set of jobs of maximum ratio among the two sets. Also, let $J^* = Max_j \cap S_j^*$ and $J = Max_j \cap S_j$ be the maximum ratio jobs in sets S_j^* and S_j respectively.

For all the jobs $i \in Max_j$, we decrease their ratio $\frac{p_{ij}}{w_i}$ by the minimum positive value Δ such that the cardinality of Max_j increases. In order to do this, we decrease the processing time of each job $i \in Max_j$ by $w_i \Delta$. After a change of this sort, in order to show that the inequality may only become tighter, we want the drop of the LHS to be less than or equal to the drop of the RHS, or

$$\sum_{i \in J^*} w_i \left(w_i + \sum_{i' \in J} w_{i'} \right) \Delta \leq \frac{1}{2\phi} \sum_{i \in J} w_i \left(\sum_{i' \in J} w_{i'} \right) \Delta$$
$$+ \frac{\phi+2}{2} \sum_{i \in J^*} w_i \left(\sum_{(i' \in J^*) \wedge (w_{i'} \leq w_i)} w_{i'} \right) \Delta.$$

If we set $A = \sum_{i \in J^*} w_i$ and $B = \sum_{i \in J} w_i$ and further notice that:

$$2 \sum_{i \in J^*} w_i \left(\sum_{(i' \in J^*) \wedge (w_{i'} \leq w_i)} w_{i'} \right) = A^2 + \sum_{i \in J^*} w_i^2,$$

the inequality becomes:

$$AB + \sum_{i \in J^*} w_i^2 \leq \frac{1}{2\phi} B^2 + \frac{\phi+2}{4} \left(A^2 + \sum_{i \in J^*} w_i^2 \right),$$

which is equivalent to:

$$\left(\frac{B}{\sqrt{2\phi}} - \sqrt{\frac{\phi}{2}}A\right)^2 + \frac{2-\phi}{4}A^2 + \frac{\phi-2}{4}\sum_{i\in J^*} w_i^2 \geq 0.$$

It thus suffices to show that:

$$\left(\frac{B}{\sqrt{2\phi}} - \sqrt{\frac{\phi}{2}}A\right)^2 + \frac{2-\phi}{4}\left(A^2 - \sum_{i\in J^*} w_i^2\right) \geq 0,$$

which is true for any value of A, B and weights of jobs in J^*.

We may now assume that for any job $i \in S_j \cup S_j^*$, $\frac{p_{ij}}{w_i} = r$ for some $r \in \mathbb{R}^+$ and therefore if any set of jobs that is a subset of $S_j \cup S_j^*$ was scheduled on j, they would all have the same completion time, equal to the sum of their processing times. Thus, the inequality that we need to prove becomes:

$$\sum_{i\in S_j^*} w_i \left(p_{ij} + \sum_{i'\in S_j} p_{i'j}\right) \leq \frac{1}{2\phi}\sum_{i\in S_j} w_i \left(\sum_{i'\in S_j} p_{i'j}\right) + \frac{\phi+2}{2}\sum_{i\in S_j^*} w_i \left(\sum_{(i'\in S_j^*)\wedge(p_{i'j}\leq p_{ij})} p_{i'j}\right).$$

If we replace p_{ij} with $w_i r$ for each job i, we obtain:

$$rAB + r\sum_{i\in J^*} w_i^2 \leq \frac{1}{2\phi}rB^2 + r\frac{\phi+2}{4}\left(A^2 + \sum_{i\in J^*} w_i^2\right),$$

which we already know is true. Now that we have proved that the initial inequality is true for all machines j, summing up the corresponding inequalities over all $j \in M$ gives us:

$$\sum_{i\in N} w_i c_i^{PS}(s_{-i}, s_i^*) \leq \frac{1}{2\phi}C^{PS}(s) + \frac{\phi+2}{2}C^{SR}(s^*). \tag{1}$$

This shows that this coordination mechanism is $\left(\frac{\phi+2}{2}, \frac{1}{2\phi}\right)$-smooth, therefore showing an upper bound of $\phi + 1 \approx 2.618$ for its robust price of anarchy and proving the theorem.

In order to get the pure PoA bound, we can once again assume that s is an equilibrium and s^* is an optimal configuration w.r.t. the SmithRule policy. For all players $i \in N$, we know that $w_i c_i^{PS}(s) \leq w_i c_i^{PS}(s_{-i}, s_i^*)$, thus:

$$C^{PS}(s) \leq \frac{1}{2\phi} C^{PS}(s) + \frac{\phi+2}{2} C^{SR}(s^*) \Rightarrow \frac{C^{PS}(s)}{C^{SR}(s^*)} \leq \phi + 1 = \frac{3+\sqrt{5}}{2} \approx 2.618.$$

The tightness of this bound follows from a Theorem shown in [11]. The authors present a weighted load balancing lower bound. The corresponding scenario in our setting is that all jobs have processing time equal to their weights and machines are using ProportionalSharing. This way, their lower bound can be translated into a *pure* PoA lower bound for restricted related machines and weighted jobs. □

The machine scheduling problem of minimizing the weighted average completion time for unrelated machines ($R| \,|\sum w_i c_i$) is well-studied NP-hard problem [32]. The first constant-factor approximation algorithm for this problem was developed by by Hall et al. [25] who achieved an approximation ratio of $\frac{16}{3}$, improved by Schulz and Skutella [38] to $\frac{3}{2} + \epsilon$. This result was further improved to $\frac{3}{2}$ independently by both Sethuraman and Squillante [40] and Skutella [41]. All of these algorithms are based either on LP or convex quadratic programming relaxations. The following Theorem presents a new polynomial time combinatorial constant-factor approximation algorithm for this optimization problem.

Theorem 4.6. *The PoA bound for* ProportionalSharing *can be used to design a combinatorial polynomial-time algorithm for* $R| \,|\sum w_i c_i$ *with an approximation guarantee of* 2.619.

Proof. Since the function $\frac{1}{2}\sum w_i(c_i(s) + p_{is_i})$ is a potential function for the game, pure Nash equilibria of the ProportionalSharing policy are equivalent to local optimal solutions of a Polynomial Local Search (PLS) problem in which local improvements, by moving one job from one machine to another machine, decrease the potential function. Therefore, we can use the result of [34] to show that a $(1-\epsilon)$-approximate local optimal solution can be found in time polynomial in $\frac{1}{\epsilon}$ and the size of the instance, i.e., a solution can be found for which any move of a job from one machine to another decreases the total potential function $\sum \frac{1}{2} w_i(c_i(s) + p_{is_i})$ by less than a $1-\epsilon$ factor.

Our goal is to change the previous proof for the PoA bound and show the approximation factor of 2.619. Let Φ^a be the value of the potential function in this approximate local optimum. Since this is an "exact" potential game, the fact that the potential function does not decrease by more than $\epsilon\Phi^a$ implies that if the game has reached a configuration s^a that corresponds to the local optimum above, moving any job from any machine to another machine does not decrease its completion time by more than $\epsilon\Phi^a$. The relaxed equilibrium inequalities for such a configuration s^a would then be: $w_i c_i^{PS}(s^a) \le w_i c_i^{PS}(s^a_{-i}, s_i^*) + \epsilon\Phi^a$ for each $i \in N$. Using inequality (1) from the previous proof, replacing s with s^a, gives:

$$C^{PS}(s^a) \le \frac{1}{2\phi} C^{PS}(s^a) + \frac{\phi+2}{2} C^{SR}(s^*) + n\epsilon\Phi^a,$$

from which, using the fact that $\Phi^a \le C^{PS}(s^a)$ and setting $\epsilon = \frac{\epsilon'}{2\phi n}$, yields:

$$\frac{C^{PS}(s^a)}{C^{SR}(s^*)} \le \frac{\phi(\phi+2)}{2\phi - 1 - \epsilon'} = \frac{\phi(\phi+2)}{2\phi - 1} + O(\epsilon') \le 2.619.$$

We have therefore shown that in time polynomial in $\frac{2\phi n}{\epsilon'}$ we can achieve an approximation ratio of $\frac{\phi(\phi+2)}{2\phi-1} + O(\epsilon')$. Of course, after we compute this allocation, we can reorder the jobs using SmithRule which can only improve the social cost value. □

Acknowledgements. We thank Ioannis Caragiannis and Tanmoy Chakraborty for helpful discussions and for pointing out related literature. The third author also thanks Yossi Azar for initial discussions about the subject of study of this paper.

References

[1] J. Aspnes, Y. Azar, A. Fiat, S.A. Plotkin, and O. Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *J. ACM*, 44(3):486–504, 1997.

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

[3] Y. Azar, K. Jain, and V.S. Mirrokni. (almost) optimal coordination mechanisms for unrelated machine scheduling. In *SODA*, 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. *Springer-Verlag*, 1984.

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

[7] A. Blum, M. Hajiaghayi, K. Ligett, and A. Roth. Regret minimization and the price of total anarchy. In *STOC*, pages 373–382, 2008.

[8] A. Borodin, M. Nielsen, and C. Rackoff. (incremental) priority algorithms. In *SODA*, pages 752 – 761, 2002.

[9] J. Bruno, E.G. Coffman, and R. Sethi. Scheduling independent tasks to reduce mean finishing time. *Communications of the ACM*, 17:382–387, 1974.

[10] I. Caragiannis. Efficient coordination mechanisms for unrelated machine scheduling. In *SODA*, pages 815–824, 2009.

[11] I. Caragiannis, M. Flammini, C. Kaklamanis, P. Kanellopoulos, and L. Moscardelli. Tight bounds for selfish and greedy load balancing. In *ICALP (1)*, pages 311–322, 2006.

[12] G. Christodoulou and E. Koutsoupias. The price of anarchy of finite congestion games. In *STOC*, pages 67–73, 2005.

[13] G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination mechanisms. *Theor. Comput. Sci.*, 410(36):3327–3336, 2009.

[14] R. Cole, Y. Dodis, and T. Roughgarden. How much can taxes help selfish routing? In *EC*, pages 98–107, 2003.

[15] R.W. Conway, W.L. Maxwell, and L.W. Miller. *Theory of Scheduling*. Addison-Wesley, Reading, MA, 1967.

[16] A. Czumaj and B. Vocking. Tight bounds for worst-case equilibria. In *SODA*, pages 413–420, 2002.

[17] E. Davis and J.M. Jaffe. Algorithms for scheduling tasks on unrelated processors. *J. ACM*, 28(4):721–736, 1981.

[18] C. Dürr and N.K. Thang. Non-clairvoyant scheduling games. In *SAGT*, pages 135–146, 2009.

[19] E. Even-dar, A. Kesselman, and Y. Mansour. Convergence time to nash equilibria. In *ICALP*, pages 502–513, 2003.

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

[21] L. Fleischer, K. Jain, and M. Mahdian. Tolls for heterogeneous selfish users in multicommodity networks and generalized congestion games. In *FOCS*, pages 277–285, 2004.

[22] L. Fleischer and Z. Svitkina. Preference-constrained oriented matching. In *ANALCO*, 2010.

[23] M. Gairing, T. Lucking, M. Mavronicolas, and B. Monien. Computing nash equilibria for scheduling on restricted parallel links. In *STOC*, pages 613–622, 2004.

[24] R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Ann. Discrete Math.*, 5:287–326, 1979.

[25] L.A. Hall, A.S. Schulz, D.B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Math. Oper. Res.*, 22(3):513–544, 1997.

[26] H. Hoogeveen, P. Schuurman, and G.J. Woeginger. Non-approximability results for scheduling problems with minsum criteria. In *IPCO*, pages 353–366, 1998.

[27] W.A. Horn. Minimizing average flow time with parallel machines. *Operations Research*, 21(3):846–847, 1973.

[28] O.H. Ibarra and C.E. Kim. Heuristic algorithms for scheduling independent tasks on nonidentical processors. *J. ACM*, 24(2):280–289, 1977.

[29] N. Immorlica, L. Li, V.S. Mirrokni, and A.S. Schulz. Coordination mechanisms for selfish scheduling. *Theor. Comput. Sci.*, 410(17):1589–1598, 2009.

[30] Y.A. Korilis, A.A. Lazar, and A. Orda. Achieving network optima using Stackelberg routing strategies. *IEEE/ACM Transactions on Networking*, 5(1):161–173, 1997.

[31] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *STACS*, pages 404–413, 1999.

[32] J.K. Lenstra, A.H.G. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Ann. Discrete Math.*, 4:281–300, 1977.

[33] J.Y-T. Leung, editor. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. CRC Press, LLC, 2004.

[34] J.B. Orlin, A.P. Punnen, and A.S. Schulz. Approximate local search in combinatorial optimization. *SIAM J. Comput.*, 33(5):1201–1214, 2004.

[35] T. Roughgarden. Stackelberg scheduling strategies. In *STOC*, pages 104–113, 2001.

[36] T. Roughgarden. Intrinsic robustness of the price of anarchy. In *STOC*, pages 513–522, 2009.

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

[38] A.S. Schulz and M. Skutella. Scheduling unrelated machines by randomized rounding. *SIAM J. Discrete Math.*, 15(4):450–469, 2002.

[39] P. Schuurman and T. Vredeveld. Performance guarantees of local search for multiprocessor scheduling. In *IPCO*, pages 370–382, 2001.

[40] J. Sethuraman and M.S. Squillante. Optimal scheduling of multiclass parallel machines. In *SODA*, pages 963–964, 1999.

[41] M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling. *J. ACM*, 48(2):206–242, 2001.

[42] M. Skutella and G.J. Woeginger. A ptas for minimizing the total weighted completion time on identical parallel machines. *Math. Oper. Res.*, 25(1):63–75, 2000.

[43] W. Smith. Various optimizers for single stage production. *Naval Res. Logist. Quart.*, 3(1-2):59–66, 1956.

[44] H. von Stackelberg. Marktform und Gleichgewicht. *Springer-Verlag*, 1934. English translation entitled *The Theory of the Market Economy*.

[45] T. Vredeveld. *Combinatorial approximation algorithms. Guaranteed versus experimental performance*. 2002. Ph.D. thesis.