Decentralized Utilitarian Mechanisms for Scheduling Games*

Richard Cole † José R. Correa ‡ Vasilis Gkatzelis § Vahab Mirrokni ¶
Neil Olver ‖

Abstract

Game Theory and Mechanism Design are by now standard tools for studying and designing massive decentralized systems. Unfortunately, designing mechanisms that induce socially efficient outcomes often requires full information and prohibitively large computational resources. In this work we study simple mechanisms that require only *local* information. Specifically, in the setting of a classical scheduling problem, we demonstrate local mechanisms that induce outcomes with social cost a small constant times that of the socially optimal solution. Somewhat counter-intuitively, we find that mechanisms yielding Pareto dominated outcomes may in fact enhance the overall performance of the system, and we provide a justification of these results by interpreting these inefficiencies as externalities being internalized. We also show how to employ randomization to obtain yet further improvements.

Lastly, we use the game-theoretic insights gained to obtain a new combinatorial approximation algorithm for the underlying optimization problem.

*This work was supported in part by NSF grants CCF0830516 and CCF1115849, by FONDECYT grant 1090050, and by Andreas Mentzelopoulos Scholarships for the University of Patras.

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

‡jcorrea@dii.uchile.cl. Departamento de Ingeniería Industrial, Universidad de Chile.

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

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

‖olver@math.mit.edu. Department of Mathematics, MIT.

1 Introduction

1.1 Multiagent resource allocation

The efficient allocation of resources has long been one of the main goals of Economics, Operations Research, and Computer Science. Recently, the study of *multiagent resource allocation* [15], which aims to achieve efficient allocation of resources among a set of self-interested agents, has been receiving increasing attention. Different types of resources, agent preferences, measures of efficiency, and allocation procedures give rise to different settings. One important distinction among allocation procedures, which will also play an important role in our work, is whether they are *centralized* or *decentralized*.

It is the centralized approach that has been traditionally followed, especially in Computer Science and Operations Research. Here, all related information is gathered (or elicited) by one entity which then aims to compute an allocation that optimizes some objective. Research along these lines has attempted to understand the computational complexity of such processes, drawing the boundary between what is tractable and what is not. For settings where this information is private and needs to be elicited from the users the main concern has been truthful implementation.

On the other hand, as the world becomes increasingly interconnected, a much more (geographically) distributed set of resources has become available, leading to a rising need for decentralized allocation processes. One of the main reasons is that centralized control in such distributed systems, examples of which include the Grid [33] and PlanetLab [48], severely impacts their scalability. The obvious drawback of decentralization, besides the limited communication among the allocation processes which may lead to inefficient solutions, is that such systems are more prone to strategic manipulation by their users. Our goal in this paper is to measure the inefficiency of the resource allocations that arise as equilibrium points of the induced game and to apply decentralized mechanisms aiming to minimize it.

1.2 Price of anarchy and coordination mechanisms

Given a game and a social cost function, the *price of anarchy* [46] has become a standard way to measure the deterioration due to selfish behavior. The price of anarchy of an instance of a game is simply the worst case ratio of the social cost at an equilibrium to that at the social optimum. The price of anarchy of a game is then the worst-case value of the price of anarchy over all instances of the game.

The games that arise within massive distributed systems like the ones mentioned above can take different forms but it is well known that strategic behavior by the users often leads to significant inefficiencies in the final allocation, i.e., a high price of anarchy. Since there is no centralized "benevolent dictator" to enforce the good behavior of the participating users, incentive design becomes a critical tool for achieving efficiency. To this end, several approaches have been proposed, including some approaches enforcing strategies on a fraction of users as a Stackelberg strategy [8, 45, 50, 60] and others using monetary transfers [9, 21, 31, 14]. The primary drawback of these methods is the need for global knowledge of the system: the mechanism itself is still centralized.

To alleviate this problem Chistodoulou, Koutsoupias and Nanavati [19] proposed a different approach, which they called *coordination mechanisms*. More specifically, they consider settings where users can choose (part of) which resource they are going to be allocated. A strategy profile then corresponds to an assignment of users to resources and a coordination mechanism aims to provide the incentives that lead to more efficient equilibrium allocations. What makes coordination mechanisms a purely decentralized solution for this setting is that they consist of several independent and decentralized allocation processes, one for each of the available resources. That

is, each resource has its own decentralized policy which decides how the resource will be allocated to the users that seek it. These policies eschew any form of monetary side payment and thus the manipulation of the users' incentives is instead achieved only through appropriate regulation of how much of their chosen resource they are allocated. This means that, for example, instead of requiring a player to contribute some form of payment, the policy would choose to provide a less valuable allocation of the resource it controls. We stress again that these policies only consider local information, i.e. the allocation choice is a function only of the users that demand it, thus remaining completely oblivious to the state of the rest of the system.

Given this decentralized regulation of the resource allocation, different coordination mechanisms induce different games, and thus lead to different sets of equilibrium allocations. Even for a fixed set of users and resources, two different coordination mechanisms may yield very different user costs for exactly the same strategy profile. It is these mechanism-specific costs that determine the social cost of a given equilibrium allocation. In evaluating the efficiency of these equilibrium points, one needs a well defined benchmark to compare this social cost against. The definition of the price of anarchy of the induced game would point to the social optimum with respect to these same costs (specific to the mechanism). The price of anarchy of a coordination mechanism is instead defined not as the price of anarchy of the induced game, but as the worst case ratio of the social cost at an equilibrium to the optimal social cost that could possibly be achieved by the centralized optimization approach. We will sometimes use the synonym *coordination ratio* as a reminder of this distinction.

1.3 Machine scheduling

In this paper, we study methods for the efficient allocation of a type of resource aiming to model machines or servers of some distributed system (such as the Internet) which offer a service that its users need. All the user tasks or jobs need to be scheduled on such machines and competition for these machines may lead to delays in servicing the users' demands. This model of *machine scheduling* [49] has been extensively studied since the 1950's, and it is generally considered a canonical model for studying settings related to job scheduling and processing.

In this model there are n jobs and m machines; each job must be assigned to a single machine. The processing time of a job can differ depending on the machine it is executed on; let p_{ij} denote the processing time of job j on machine i. Each job can also have an associated weight, which may be interpreted as a measure of importance. A resource allocation for such a problem instance (a *schedule*) consists of an assignment of jobs to machines, and a specification of the order according to which the jobs on each machine will be processed. For any such assignment and ordering, the *completion time* of each job is now determined; if on machine i the assigned jobs are in the order $j_1, \ldots, j_r$, then the completion time of job j_1 is p_{ij_1}, the completion time of job j_2 is $p_{ij_1} + p_{ij_2}$, and so on.

The most basic model is that of *identical* machines, where the processing time of any job is the same on all of the machines. In the more general model of *related* machines each machine has a speed, and the processing time of a job on a machine is inversely proportional to the speed of that machine. The main scheduling model that we study is *unrelated* machine scheduling in which the processing times are arbitrary, thus capturing all the above models as special cases.

In this work we consider the *scheduling game* that is induced due to the lack of centralized control. Each job is a fully informed player wanting to minimize its individual completion time, and its set of strategies correspond to the set of machines. A job's completion time on a machine depends not only on the strategies chosen by other players (in particular, which other players chose

that machine), but also on the order that the jobs are run on the machine[1]; in other words, this is a setting with *externalities*. The cost of a job will be its weighted completion time; its completion time multiplied by its weight. The objective function that research on this setting has mostly focused on is the *makespan*, i.e. the maximum completion time over all jobs, which, for unweighted jobs, corresponds to the egalitarian social cost (see Section 1.5). This paper studies the utilitarian social cost instead, i.e. the unweighted or weighted sums of completion times.

A coordination mechanism for this setting is a set of *local policies*, one per machine, specifying how the jobs assigned to that machine are scheduled. We will actually consider the slightly more restrictive class of *strongly local policies*. For such policies the schedule on machine i must be a function of only the processing times p_{ij} and weights w_j of the jobs assigned to the machine. In contrast, in simply *local policies* the schedule on a machine may in addition depend on the full vector $\boldsymbol{p}_j = (p_{1j}, p_{2j}, \ldots, p_{mj})$ of processing times of each job j assigned to the machine. This is a somewhat weaker notion of locality, providing the policies with more information about the given problem instance.

1.4 Our Results

We begin by studying Smith's Rule, a policy according to which machines process jobs in increasing order of their processing time to weight ratio. This is a natural first candidate to analyze since it is known that, for any given assignment of jobs to machines, this is the policy that minimizes our social cost function [59]. We prove that the coordination ratio for this policy is exactly 4, improving upon a result by Correa and Queyranne [24], who showed the same bound but for the less general model of restricted related machines (see Section 2). The constant coordination ratio for the weighted sum of completion times is in sharp contrast to the negative results that have been shown for the makespan objective, for which no natural coordination mechanism can achieve a constant coordination ratio [1] (See Section 1.5).

We also show that if we restrict ourselves to deterministic policies which always run jobs one after the other in some order, regardless of how this ordering depends on the weights and processing times of the assigned jobs—then this factor of 4 cannot be improved. We overcome this barrier in two ways; the first is deterministic, and adds artificial delays; the second is randomized, and achieves an even better total welfare.

Among them, the deterministic policy is most naturally described as a *preemptive* one. In our context, preemption really refers to time multiplexing: the machine runs the jobs "in parallel", dividing its processing resources amongst the active jobs. The preemptive policy that we consider (ProportionalSharing) splits the processing capacity of a machine among its uncompleted jobs in proportion to their weights. This generalizes the EqualSharing policy [27], which splits the processing capacity equally amongst the jobs; what ProportionalSharing does for the unweighted case. We uncover a close connection of this policy to Smith's Rule, allowing us to apply a similar proof strategy, but yielding a significantly improved coordination factor of 2.618. This improvement using preemption is somewhat counter-intuitive if one considers the fact that, for any preemptive policy, there is a non-preemptive one that Pareto dominates it for any given assignment of jobs to machines. This result is also in contrast to the makespan case, where even in the unweighted case the EqualSharing policy achieves a coordination ratio of $\Theta(m)$ [27], no better than Smith's rule. To make sense of this phenomenon we show that, for a fixed assignment, the cost that each job suffers according to this very natural preemptive policy actually equals the cost that it would suffer if Smith's Rule were instead being used, plus the externalities that this job would cause

[1] Actually, we will allow schedules that are more general than just executing the jobs in some order, but this simplifies the discussion for now.

to the jobs that would have been scheduled after it. Each job is therefore forced to *internalize externalities* that it causes to jobs on the same machine, leading to improved incentives. We also show an improved bound of 2.5 for the coordination ratio in the unweighted case.

On the other hand, we show that, under some restrictions, no deterministic policy can achieve a factor better than 2.166. To break this new barrier we consider a policy we call Rand, in which jobs are randomly (but non-uniformly) ordered, based on their processing time to weight ratio. This randomized policy also forces jobs which have a higher priority according to Smith's Rule to suffer delays due to externalities that they cause, again leading to better incentives and even more efficient equilibrium allocations. One of the benefits of randomization is that although the jobs are made to suffer for (part of) their externalities, the schedule that the policy produces is always Pareto efficient. We give a bound of $32/15 \approx 2.133$ for the coordination ratio of this policy, a significant improvement over ProportionalSharing. In addition, in the case where the weighted sum of processing times is negligible compared to the total cost, our randomized policy has a much better coordination ratio of $\pi/2$, which is tight. The proofs here are perhaps the most interesting, involving a connection to the classical *Hilbert matrix*.

We prove all of the upper bound results in a common framework that brings out the structure in the scheduling games we consider. Once the framework has been set up, our proofs become short and elegant, and we anticipate that the approach may prove useful elsewhere too. We are able to relate the games induced by each of the policies we consider to certain inner product spaces. Proving upper bounds on the price of anarchy then becomes much simpler, in most cases involving an application of Cauchy-Schwartz and some form of "norm distortion" inequality to relate back to the Smith's rule cost. While we present our proofs for pure strategies and pure Nash equilibria, we observe that all the results can be stated within the smoothness framework of Roughgarden [51] (see Section 2). This implies that all the bounds hold for more general equilibrium concepts including mixed Nash equilibria and correlated equilibria.

The game obtained when using Smith's rule as the policy has a defect: it does not necessarily possess pure Nash equilibria [24]. Nevertheless, we show that the other policies we consider all induce exact potential games, giving another indication that ProportionalSharing and Rand are very natural policies. In fact, we can use these properties, along with other game-theoretic insights we have gained to give a result for the underlying *centralized* optimization problem.

From a purely centralized optimization perspective, the problem of minimizing the weighted sum of completion times has been extensively studied. The problem is APX-hard [40] on unrelated machines, and the current best polynomial time algorithm has an approximation factor of $\frac{3}{2}$ [55, 57]. All previous constant-factor approximation algorithms are based on rounding linear or convex programs. Complementing all these known non-combinatorial approximation algorithms, we design a new *combinatorial* $(2+\epsilon)$-approximation algorithm for optimizing the weighted sum of completion times on unrelated machines.

In designing our approximation algorithm we take advantage of the fact that the best-response dynamics of the induced game are related to local search algorithms. Starting from an initial solution, a local search algorithm iteratively moves to neighboring solutions which improve the global objective. This is based on a neighborhood relation that is defined on the set of solutions. Now, if one considers the strategy profiles of the game induced by the coordination mechanism as solutions, the best-response moves of the users in this game implicitly define the set of possible local moves. The speed of convergence and the approximation factor of local search algorithms for scheduling problems have been studied mainly for the makespan objective function [26, 28, 30, 42, 52, 54, 61, 2, 7]. Our combinatorial approximation algorithm for the weighted sum of completion times is the first local search algorithm for this problem, and is different from the previously studied

algorithms for the makespan objective. The neighborhood implicitly defined by the coordination mechanism at hand is non-trivial and it seems unlikely that such a simple algorithm could be designed without the initial game-theoretic intuition.

1.5 Related Work

Previous work on scheduling games mainly concerned the makespan social cost. One of the first such games to be considered was the one induced by the Makespan policy [46], according to which all jobs are released at the same time; each job's completion time is equal to the sum of the processing times of the jobs assigned to its machine. This scheduling game gathered significant attention, eventually leading to a sequence of tight price of anarchy bounds for different machine models [25, 34, 5]. The games induced by Makespan are also known as *load balancing games*. In their paper introducing coordination mechanism design [19], Christodoulou, Koutsoupias and Nanavati analyzed mechanisms for identical machines using the ShortestFirst and LongestFirst policies, which process jobs in non-decreasing and non-increasing order of their processing times respectively. Immorlica et al. [43] studied these three coordination mechanisms, along with a randomized one which orders jobs randomly in a uniform fashion, for several different machine scheduling models. They surveyed the known results for these settings, uncovering connections of these local policies with greedy and local search algorithms [42, 30, 52, 26, 2, 7, 10, 61]. Apart from price of anarchy related results, they also studied the speed of convergence to equilibria and the existence of pure Nash equilibria for the ShortestFirst and LongestFirst policies. Azar, Jain, and Mirrokni [6] showed that the ShortestFirst policy and in fact any *strongly* local fixed ordering policy (defined in Section 2) does not achieve a coordination ratio better than $\Omega(m)$. Additionally, they presented a non-preemptive local policy that achieves a coordination ratio of $O(\log m)$ and a policy that induces potential games and gives a coordination ratio of $O(\log^2 m)$. Caragiannis [12], among other results, showed an alternative coordination mechanism that guaranteed a coordination ratio of $O(\log m)$ for unrelated machines, while still inducing potential games. Fleischer and Svitkina [32] showed a lower bound of $\Omega(\log m)$ for all local fixed ordering policies, thus proving that Caragiannis' mechanism is optimal with respect to the price of anarchy within this class. This bound had though already been overcome by Caragiannis [12] who presented a local preemptive policy with an approximation factor of $O(\log m/\log\log m)$. Very recent work by Abed and Huang [1] showed that this factor is the best that can be achieved by any natural policy, including preemptive and randomized ones.

Our work concerns the utilitarian social cost, or (weighted) sum of completion times. For this objective, Correa and Queyranne [24] studied Smith's rule for the restricted related machine model and they exhibited an instance for which the induced game does not possess a pure Nash equilibrium. They also presented bounds for the price of anarchy of SmithRule in this model. Finally, Hoeksma and Uetz [39] showed better price of anarchy bounds for the less general setting of unweighted jobs and related machines using ShortestFirst; the unweighted variant of SmithRule.

2 Preliminaries

Throughout this paper, let J be a set of n jobs to be scheduled on a set I of m machines. Let p_{ij} denote the processing time of job $j \in J$ on machine $i \in I$ and w_j denote its weight (or importance). The shorthand notation ρ_{ij} will be used for the ratio p_{ij}/w_j. Jobs that have both the same processing time and the same weight can be distinguished from one another only if they have been assigned a unique ID; otherwise, the jobs are called *anonymous*.

We will refer to the following standard scheduling models:

Identical machines. All machines are identical, meaning each job needs the same processing time on each machine: $p_{ij} = p_{i'j}$ for all $i, i' \in I$. The model of *restricted* identical machines is a variant according to which each job can be run only on some specified subset of machines.

Related machines. The machines may have different speeds, and the processing time of a job is inversely proportional to the speed: $p_{ij} = p_j/\sigma_i$, where σ_i represents the speed of machine i, and p_j the processing requirement of job j. The *restricted* related machines variant is again obtained by possibly restricting the set of machines to which each job can be assigned.

Unrelated machines. The processing times are arbitrary. This is the most general of these models. There is no need to distinguish between restricted and unrestricted variants, since we allow specifying that a job takes infinite time on a machine.

A *coordination mechanism* for this setting is a set of local policies, one for each machine. Each such policy determines how to schedule the set of jobs assigned to the machine it controls, thus defining the *completion time* c_j of each job j in that set. A coordination mechanism thereby gives rise to a *scheduling game* in which there are n agents (jobs) and each agent's strategy set is the set of machines I. A strategy profile (or configuration) corresponds to an assignment of jobs to machines, represented by a vector $\boldsymbol{x}$, where x_j gives the machine to which job j is assigned. Given such an assignment $\boldsymbol{x}$, the cost of job j is its weighted completion time, as determined by the policy on the machine x_j. We let $w_j c_j^\alpha(\boldsymbol{x})$ and $C^\alpha(\boldsymbol{x})$ denote the cost for player j and the social cost respectively, where $\alpha \in \{SR, PS, SF, ES, R\}$ denotes the policy, namely SmithRule, ProportionalSharing, ShortestFirst, EqualSharing and Rand, respectively.[2] The agent controlling each job aims to choose a strategy (i.e., a machine) that minimizes its cost or, in the case of randomized policies, its expected cost. The mechanisms that we analyze are designed with the goal of minimizing the utilitarian social cost, i.e. $C^\alpha(\boldsymbol{x}) = \sum_{j \in J} w_j c_j^\alpha(\boldsymbol{x})$.

A strategy profile $\boldsymbol{x}$ of a scheduling game instance is a *pure Nash equilibrium* (PNE) if no player has an incentive to unilaterally change its strategy. Formally, if this instance is induced by coordination mechanism α, then for all $j \in J$ and all $i \in I$ we get $c_j^\alpha(\boldsymbol{x}) \leq c_j^\alpha(\boldsymbol{x}_{-j}, i)$, where $(\boldsymbol{x}_{-j}, i)$ denotes the assignment $\boldsymbol{x}$, except modified so that job j is assigned to machine i.

In order to measure the efficiency of a coordination mechanism α for a given scheduling game instance, we study its social cost in PNE assignments. We are interested in the worst case ratio of the social cost in a PNE assignment divided by the optimal social cost achievable from a centralized optimization approach. It is known that the optimal solution of the centralized optimization problem schedules jobs on machines according to SmithRule [59], so the optimal social cost can be expressed as $C^{SR}(\boldsymbol{x}^*)$, where $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x'}} C^{SR}(\boldsymbol{x'})$. If we also let $E(\alpha)$ be the set of equilibria induced by α, and $\boldsymbol{x} = \arg\max_{\boldsymbol{x'} \in E(\alpha)} C^\alpha(\boldsymbol{x'})$ be the worst equilibrium assignment with respect to the social cost induced by α, then this ratio is equal to $C^\alpha(\boldsymbol{x})/C^{SR}(\boldsymbol{x}^*)$ for the given game instance. Following the definition of [19] the *(pure) price of anarchy* or *coordination ratio* of coordination mechanism α is defined to be the maximum such ratio, taken over all the scheduling game instances that the mechanism may induce. Slightly abusing notation, we use $X_i = \{j \in J \mid x_j = i\}$ to denote the set of jobs allocated to machine i in configuration $\boldsymbol{x}$, and X_i^* analogously for $\boldsymbol{x}^*$.

Adapting the work of Roughgarden [51] to this setting, we define a coordination mechanism α to be (λ, μ)-smooth if for any two configurations $\boldsymbol{x}$ and $\boldsymbol{x'}$,

$$\sum_{j \in J} w_j c_j^\alpha(\boldsymbol{x}_{-j}, x'_j) \leq \lambda C^{SR}(\boldsymbol{x'}) + \mu C^\alpha(\boldsymbol{x}).$$

[2]The coordination mechanisms we study in this paper use the same local policy on each machine, so henceforth we refer to a coordination mechanism using the name of the policy.

If a coordination mechanism is (λ, μ) smooth, then this yields an upper bound of $\frac{\lambda}{1-\mu}$ which applies not only to its pure price of anarchy, but also to its *robust price of anarchy*. This, among other things, implies that it is not only the social cost of PNE that is bound to be at most $\frac{\lambda}{1-\mu}$ times the optimal social cost, but also the social cost of any correlated equilibrium [51].

A game is a *potential game* if there exists a potential function over the set of strategy profiles such that any player's unilateral deviation leads to a drop of the potential function if and only if that player's cost drops. A potential game is *exact* if after each move, the changes to the potential function and to the player's cost are equal. It is easy to see that a potential game always possesses a pure Nash equilibrium, corresponding to a local minimum of the potential function.

2.1 Classification of policies

It will be useful (particularly for discussing lower bounds) to identify the main classes of strongly local policies that will concern us in this work.

Fixed ordering policies[3]**.** These policies assign an order on *all* jobs, based on the jobs' characteristics on the machine (processing time, weight, and possibly ID). Then, for a given assignment, the jobs assigned to the machine are executed according to this order. One motivation for these policies is that they satisfy the *independence of irrelevant alternatives (IIA)* property: for any pair of jobs, their relative ordering is independent of which other jobs are assigned to the machine. This property appears as an axiom in voting theory, bargaining theory and logic [56].

Flexible ordering policies. In this class, policies still execute jobs in some fixed order, but that order may depend arbitrarily on the set of jobs assigned to the machine. Here we require that the jobs on a machine are executed consecutively in some deterministic order. Moreover, we require that there be no idle time between jobs, and that jobs are released immediately upon completion. The reason for this restriction is to distinguish from the next case.

Preemptive. Preemption refers to the ability to suspend a job before it completes in order to execute another job.[4] The initial job can then be resumed later. Preemption allows for *time multiplexing*: by switching between a number of jobs very quickly, the illusion is given that the jobs are being run simultaneously on the machine. Preemptive policies can also introduce idle time intervals during which no job is being processed (e.g. [12]); we call the ones that do not *prompt*. In fact, any preemptive policy yields a schedule which is Pareto dominated by that of some policy that does not use preemption. Thus, as we explain in more detail in Section 4, such policies can equally well be considered as flexible ordering policies, but where jobs may be held back after completion.

Randomized. Here, the policy may schedule the jobs randomly, according to a distribution depending only on the processing times and weights of the jobs on the machine. While more general schedules are possible, it's helpful to think of simply a random ordering of the jobs.

We also call, e.g., a coordination mechanism consisting of fixed ordering policies a "fixed ordering coordination mechanism".

[3]These were called simply *ordering policies* in [6], but we wish to emphasize the distinction with the superset of flexible ordering policies, defined next.

[4]Note that, unlike in some literature on machine scheduling, preemption here does *not* imply that a job can be processed on a different machine after it is suspended.

3 Smith's Rule

Smith's rule is a fixed ordering policy that schedules jobs on machine i in increasing order of $\rho_{ij} = p_{ij}/w_j$. In the unweighted case, this reduces to the ShortestFirst policy. It is known that given an assignment of jobs to machines, in order to minimize the weighted sum of completion times, using Smith's rule is optimal [59]. It is therefore natural to consider this policy as a good first candidate to study. Our first theorem shows that using this rule will result in Nash equilibria with social cost at most a constant factor of 4 away from the optimum.

Our price of anarchy related proofs will use a common framework. The proof for Smith's rule is the simplest, but it will introduce a number of aspects of this framework. In the following two proofs, for notational simplicity we assume that all jobs assigned to the same machine have distinct ratios (of processing time to weight) on that machine.[5] Also, we index some of the intermediate inequalities in the derivations of these proofs in order to refer to them in subsequent discussion.

We will construct a mapping from the set of configurations to a certain inner product space, such that the norm of the mapping will closely correspond to the cost of the configuration. To wit, define the map $\varphi : I^J \to L_2([0,\infty))^I$, which maps every strategy profile $\boldsymbol{x}$ to a vector of functions (one for each machine) as follows. If $\boldsymbol{f} = \varphi(\boldsymbol{x})$, then for each $i \in I$

$$f_i(y) = \sum_{j \in X_i : \rho_{ij} \geq y} w_j \qquad \text{(recall that } \rho_{ij} = p_{ij}/w_j\text{)}.$$

Notice that $f_i(\rho_{ij})$, multiplied by p_{ij}, is simply the marginal social cost due to job j, i.e., its own cost plus the cost it induces on other jobs. We let $\langle f, g \rangle := \int_0^\infty f(y)g(y)dy$ denote the usual inner product on L_2, and in addition define $\langle \boldsymbol{f}, \boldsymbol{g} \rangle := \sum_{i \in I} \langle f_i, g_i \rangle$. In both cases, $\|\cdot\|$ refers to the induced norm. Next, define $\eta(\boldsymbol{x})$ to be the weighted processing time of all the jobs:

$$\eta(\boldsymbol{x}) = \sum_{j \in J} w_j p_{x_j j}.$$

We can then write the cost of a configuration in terms of $\varphi(\boldsymbol{x})$ and $\eta(\boldsymbol{x})$:

Lemma 3.1. *For any configuration $\boldsymbol{x}$, $C^{SR}(\boldsymbol{x}) = \frac{1}{2}\|\varphi(\boldsymbol{x})\|^2 + \frac{1}{2}\eta(\boldsymbol{x})$.*

[5] The proofs can be adapted to the case of non-distinct ratios by replacing the condition $\rho_{ik} < \rho_{ij}$ that appears in the terms of sums with the condition $\rho_{ik} \leq \rho_{ij}$, and introducing a tie breaking rule.

Proof. Let $\boldsymbol{f} = \varphi(\boldsymbol{x})$. We have

$$\begin{aligned}
\|\varphi(\boldsymbol{x})\|^2 &= \sum_{i\in I}\int_0^\infty f_i(y)^2 dy \\
&= \sum_{i\in I}\int_0^\infty \sum_{\substack{j\in X_i\\ \rho_{ij}\geq y}} w_j \sum_{\substack{k\in X_i\\ \rho_{ik}\geq y}} w_k dy \\
&= \sum_{i\in I}\sum_{j\in X_i}\sum_{k\in X_i} w_j w_k \int_0^\infty \mathbf{1}_{\rho_{ij}\geq y}\mathbf{1}_{\rho_{ik}\geq y} dy \\
&= \sum_{i\in I}\sum_{j\in X_i}\sum_{k\in X_i} w_j w_k \min\{\rho_{ij},\rho_{ik}\} \qquad (1)\\
&= \sum_{i\in I}\sum_{j\in X_i} w_j\Big(\, 2\sum_{\substack{k\in X_i\\ \rho_{ik}<\rho_{ij}}} w_k\rho_{ik} \;+\; w_j\rho_{ij}\Big) \\
&= \sum_{i\in I}\sum_{j\in X_i} w_j\Big(\, 2\sum_{\substack{k\in X_i\\ \rho_{ik}\leq\rho_{ij}}} p_{ik} \;-\; p_{ij}\Big) \\
&= 2C^{SR}(\boldsymbol{x}) - \eta(\boldsymbol{x}).
\end{aligned}$$

The result follows. □

Theorem 3.2. *The price of anarchy of* SmithRule *for unrelated machines is at most* 4.

Proof. Let $\boldsymbol{x}$ and $\boldsymbol{x^*}$ be two assignments, with $\boldsymbol{x}$ being a Nash equilibrium, and write $\boldsymbol{f} = \varphi(\boldsymbol{x})$, and $\boldsymbol{f^*} = \varphi(\boldsymbol{x^*})$. We first calculate a job j's completion time according to $\boldsymbol{x}$, and then we use the Nash condition that $c_j^{SR}(\boldsymbol{x}) \leq c_j^{SR}(\boldsymbol{x}_{-j}, x_j^*)$ for every job j.

$$c_j^{SR}(\boldsymbol{x}) = \sum_{\substack{k\in J:\ x_k=x_j\\ \rho_{x_jk}<\rho_{x_jj}}} p_{x_jk} \;+\; p_{x_jj} \leq \sum_{\substack{k\in J:\ x_k=x_j^*\\ \rho_{x_j^*k}<\rho_{x_j^*j}}} p_{x_j^*k} \;+\; p_{x_j^*j}.$$

So
$$\begin{aligned}
C^{SR}(\boldsymbol{x}) = \sum_{j\in J} w_j c_j^{SR}(\boldsymbol{x}) &\leq \sum_{i\in I}\sum_{j\in X_i^*} w_j\Big(\sum_{\substack{k\in X_i\\ \rho_{ik}<\rho_{ij}}} p_{ik} + p_{ij}\Big) \\
&\leq \sum_{i\in I}\sum_{j\in X_i^*}\Big(\sum_{\substack{k\in X_i\\ \rho_{ik}<\rho_{ij}}} w_j w_k\rho_{ik} + w_j p_{ij}\Big) \\
&\leq \sum_{i\in I}\sum_{j\in X_i^*}\sum_{k\in X_i} w_j w_k \min\{\rho_{ik},\rho_{ij}\} \;+\; \sum_{i\in I}\sum_{j\in X_i^*} w_j p_{ij} \qquad (2)\\
&= \sum_{i\in I}\sum_{j\in X_i^*}\sum_{k\in X_i} w_j w_k \int_0^\infty \mathbf{1}_{\rho_{ij}\geq y}\mathbf{1}_{\rho_{ik}\geq y} dy \;+\; \eta(\boldsymbol{x^*}) \\
&= \sum_{i\in I}\int_0^\infty \sum_{\substack{j\in X_i^*\\ \rho_{ij}\geq y}} w_j \sum_{\substack{k\in X_i\\ \rho_{ik}\geq y}} w_k dy \;+\; \eta(\boldsymbol{x^*}) \\
&= \sum_{i\in I}\int_0^\infty f_i(y) f_i^*(y) dy \;+\; \eta(\boldsymbol{x^*}) \\
&= \langle \boldsymbol{f}, \boldsymbol{f^*}\rangle \;+\; \eta(\boldsymbol{x^*}). \qquad (3)
\end{aligned}$$

Now applying Cauchy-Schwartz, followed by the inequality $ab \leq a^2 + b^2/4$ for $a, b \geq 0$, we obtain

$$\begin{aligned} C^{SR}(\boldsymbol{x}) &\leq \|\boldsymbol{f}\|\|\boldsymbol{f}^*\| + \eta(\boldsymbol{x}^*) \\ &\leq \|\boldsymbol{f}^*\|^2 + \tfrac{1}{4}\|\boldsymbol{f}\|^2 + \eta(\boldsymbol{x}^*) \\ &\leq 2C^{SR}(\boldsymbol{x}^*) + \tfrac{1}{2}C^{SR}(\boldsymbol{x}) \qquad \text{by Lemma 3.1.} \end{aligned}$$

Hence $C^{SR}(\boldsymbol{x}) \leq 4C^{SR}(\boldsymbol{x}^*)$. □

Notice that in this proof, the cost $C^{SR}(\boldsymbol{x})$ of an assignment $\boldsymbol{x}$ is closely related to the norm of $\varphi(\boldsymbol{x})$, and the inequality obtained from the Nash condition is bounded by a term involving the inner product $\langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}^*) \rangle$. This will be a common feature of all our proofs.

For simplicity, the proof above was written as a pure price of anarchy bound; $\boldsymbol{x}$ was taken to be a pure Nash equilibrium. However, it is clear that the proof in fact yields a robust price of anarchy bound, as defined by Roughgarden [51]. More precisely, the above proof shows that SmithRule is $(2, 1/2)$-smooth.

The following result, proved in Appendix A, shows that no fixed ordering coordination mechanism can do better than SmithRule. (In fact, the result can be extended to all *flexible* ordering coordination mechanisms, but we will not discuss this here.) This also implies that the bound of Theorem 3.2 is tight.

Theorem 3.3. *The pure price of anarchy of any set of fixed ordering policies is at least 4. This is true even for the case of restricted identical machines with unweighted jobs.*

We note that for the unit weight case, a constant upper bound on the coordination ratio of Smith's rule can be obtained via a reduction from the *priority routing model* of Farzad et al. [29]. However, the resulting bound is not optimal.

4 Improvements with Preemption and Randomization

4.1 Preemptive Coordination Mechanism

In this section, we study the power of preemption (or equivalently, delays) and present the following preemptive policy, named ProportionalSharing. Jobs are scheduled in parallel using time-multiplexing, and, at any moment in time, each uncompleted assigned job receives a fraction of the processor time equal to its weight divided by the total weight of uncompleted jobs on the machine. In the unweighted case, this gives the EqualSharing policy.

We will show that ProportionalSharing has a better coordination ratio than any fixed ordering policy. These results create a clear dichotomy between such policies and ProportionalSharing. This may seem counter-intuitive at first, since, given an assignment of jobs to machines, the schedule produced by ProportionalSharing is Pareto dominated by that of SmithRule. To be more precise, on each machine, every job apart from the one that SmithRule would schedule last, strictly prefers SmithRule to ProportionalSharing; the one scheduled last is indifferent between the two schedules. This can also be seen in Figure 1 which compares how the two policies would schedule a given set of jobs on the same machine.

Lemma 4.1. *Given an assignment $\boldsymbol{x}$, the weighted completion time of a job j on some machine i using ProportionalSharing (whether currently assigned there or not) is*

$$w_j c_j^{PS}(\boldsymbol{x}_{-j}, i) = \sum_{k \in X_i \setminus \{j\}} w_j w_k \min\{\rho_{ij}, \rho_{ik}\} \; + \; w_j p_{ij}.$$

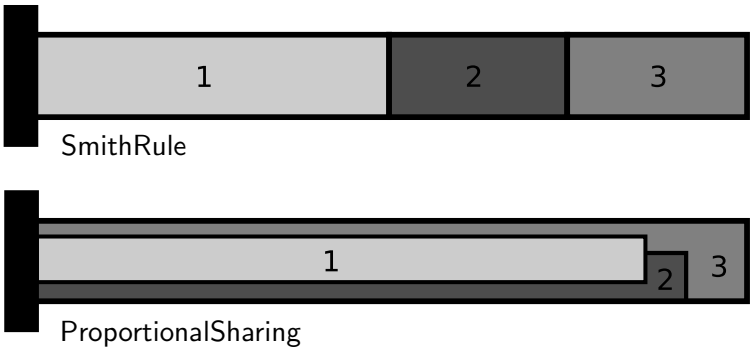

Figure 1: Three jobs scheduled on some machine i, which uses SmithRule in the first case and ProportionalSharing in the second. Their processing times and weights are $p_{i1} = 4$ and $w_1 = 7$ for the first job, $p_{i2} = 2$ and $w_2 = 3$ for the second, and $p_{i3} = 2$ and $w_3 = 2$ for the third.

Proof. First, observe that for two jobs k and k' with $\rho_{ik} \leq \rho_{ik'}$, job k will complete before (or at the same time as) job k' when ProportionalSharing is used. To see this, consider the situation at the time when the earlier of the two jobs is completed. Let q and q' be the amount of processing time that has been allocated to k and k' by this time. Then $q' = \frac{w_{k'}}{w_k} q$. If k is not completed, then $q < w_k \rho_{ik}$, and so $q' < w_{k'} \rho_{ik} \leq p_{ik'}$, and k' is not completed either.

Let t be the time when job j is completed. All jobs k with $\rho_{ik} \leq \rho_{ij}$ have completed by this time; thus each such job has received p_{ik} units of processing time. On the other hand, all jobs k with $\rho_{ik} > \rho_{ij}$ are not yet complete at time t, and for each w_j units of processing time job j receives, job k receives w_k units. Thus by time t, the processing time spent on any such job k will be exactly $\frac{p_{ij} w_k}{w_j}$. Since the total processing time is the sum of the processing times allocated to all the jobs, we thus have that

$$t = \sum_{\substack{k \in X_i \setminus \{j\} \\ \rho_{ik} \leq \rho_{ij}}} p_{ik} \; + \sum_{\substack{k \in X_i \\ \rho_{ik} > \rho_{ij}}} \frac{w_k}{w_j} p_{ij} \; + \; p_{ij}$$

Thus

$$\begin{aligned} w_j c_j^{PS}(\boldsymbol{x}_{-j}, i) &= \sum_{\substack{k \in X_i \setminus \{j\} \\ \rho_{ik} \leq \rho_{ij}}} w_j p_{ik} \; + \sum_{\substack{k \in X_i \\ \rho_{ik} > \rho_{ij}}} w_k p_{ij} \; + \; w_j p_{ij} \\ &= \sum_{k \in X_i \setminus \{j\}} w_j w_k \min\{\rho_{ij}, \rho_{ik}\} \; + \; w_j p_{ij}. \end{aligned}$$

□

A better understanding of why ProportionalSharing performs better despite the Pareto inefficiency of the schedules it produces can be obtained by observing the following corollary of Lemma 4.1.

Corollary 4.2. *Given an assignment $\boldsymbol{x}$, the weighted completion time of a job j on some machine i using ProportionalSharing is*

$$w_j c_j^{PS}(\boldsymbol{x}_{-j}, i) = w_j c_j^{SR}(\boldsymbol{x}_{-j}, i) + \sum_{\substack{k \in X_i \\ \rho_{ik} > \rho_{ij}}} w_k p_{ij}.$$

This corollary precisely quantifies what cost, in addition to the SmithRule cost, this job is forced to suffer. A closer look reveals that this additional cost (the rightmost term) is exactly equal to externalities that job j would cause if the assignment of jobs to machines remained the same but SmithRule was used instead. That is, the sum for each job k that would have been scheduled after job j ($\rho_{ik} > \rho_{ij}$), of the cost increase that job j causes to that job ($w_k p_{ij}$). From this perspective, ProportionalSharing can be thought of (and also implemented) as using SmithRule to determine the processing order, but then delaying the release of each job after it is completed until the additional cost equals these externalities. Since we already know that, for any given assignment, SmithRule would produce the social welfare maximizing schedule, one may expect that our preemptive policy exactly "internalizes the externalities" of the players and should therefore lead to the optimal assignment in equilibrium. The reason why this is not the case is that the participation of job j in the game does not cause externalities only to jobs that are assigned to its machine. Nevertheless, our policies are necessarily oblivious to what the state of the system is beyond the machine they control, so these "local externalities" may be the best possible alternative. By taking these local externalities into consideration, ProportionalSharing better aligns the interests of a player with those of the system, leading not only to better assignments than SmithRule but also to a better social cost, despite the (otherwise unnecessary) delays suffered. Another perspective on the delays is that they are a form of money that the players are forced to pay, but this is money that can only be "burned" and not transfered. From this perspective, our setting is similar to that of *money burning mechanisms* [37], with the added restriction that the "payments" have to be a function of only local information. These two restrictions preclude the implementation of welfare-maximizing mechanisms like VCG, but nonetheless our mechanisms define payments that lead to surprisingly low social cost.

From Lemma 4.1 and (1) we also immediately obtain the following corollary (note the factor 2 difference compared to the main term in the cost of Smith's rule in Lemma 3.1), which will be used in proving the two subsequent theorems:

Corollary 4.3. *For any assignment $\boldsymbol{x}$, $C^{PS}(\boldsymbol{x}) = \|\varphi(\boldsymbol{x})\|^2$.*

Theorem 4.4. *The price of anarchy of ProportionalSharing for unrelated machines is at most $\phi+1 = \frac{3+\sqrt{5}}{2} \approx 2.618$. Moreover, this bound is tight even for the restricted related machines model.*

Proof. Let $\boldsymbol{x}$ be an equilibrium assignment, and $\boldsymbol{x}^*$ any arbitrary assignmnent. From the Nash condition,

$$\begin{aligned} C^{PS}(\boldsymbol{x}) &\leq \sum_{j\in J} w_j c^{PS}(\boldsymbol{x}_{-j}, x_j^*) \\ &\leq \sum_{i\in I} \sum_{j\in X_i^*} \left(\sum_{k\in X_i} w_j w_k \min\{\rho_{ij}, \rho_{ik}\} + w_j p_{ij} \right) && \text{by Lemma 4.1} \\ &= \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}^*) \rangle + \eta(\boldsymbol{x}^*) && \text{by steps (2)-(3)} \end{aligned} \tag{4}$$

Following the same method of analysis as for Smith's Rule, we obtain

$$\begin{aligned} C^{PS}(\boldsymbol{x}) &\leq \|\varphi(\boldsymbol{x})\| \|\varphi(\boldsymbol{x}^*)\| + \eta(\boldsymbol{x}^*) \\ &\leq \alpha \|\varphi(\boldsymbol{x}^*)\|^2 + \tfrac{1}{4\alpha} \|\varphi(\boldsymbol{x})\|^2 + \eta(\boldsymbol{x}^*) \\ &\leq 2\alpha C^{SR}(\boldsymbol{x}^*) + \tfrac{1}{4\alpha} C^{PS}(\boldsymbol{x}) + (1-\alpha)\eta(\boldsymbol{x}^*) \\ &\leq (1+\alpha) C^{SR}(\boldsymbol{x}^*) + \tfrac{1}{4\alpha} C^{PS}(\boldsymbol{x}), \end{aligned}$$

using Cauchy-Schwartz inequality and the fact that $\eta(\boldsymbol{x}^*) \leq C^{SR}(\boldsymbol{x}^*)$. Setting $\alpha = (1+\sqrt{5})/4$ yields $C^{PS}(\boldsymbol{x})/C^{SR}(\boldsymbol{x}^*) \leq \frac{3+\sqrt{5}}{2}$.

The tightness of this bound follows from a construction in [13], where in fact they show that even if C^{PS} is used as the benchmark, i.e., we consider the ratio $C^{PS}(\boldsymbol{x})/C^{PS}(\boldsymbol{x}^*)$, this can be arbitrarily close to $1+\phi$. □

The reader will observe how similar the proof above was to the proof of Theorem 3.2, once the relevant costs have been described in terms of the inner product. In particular, (4) is obtained by following precisely the same steps followed in the proof of Theorem 3.2 to get from (2) to (3). In that proof, (2) can be interpreted as a kind of symmetrization step, which is needed since the inner product $\langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}^*) \rangle$ is symmetric. ProportionalSharing is already symmetric in the appropriate sense, and so there is a tighter connection between the Nash condition and the inner product. This same symmetry property will be shared in the randomized policy we consider in the next section.

Notice also that since SmithRule and ProportionalSharing were described in terms of the same mapping and inner product, it was very easy to relate $\|\varphi(\boldsymbol{x}^*)\|$ back to the cost of SmithRule. This will be less straightforward for the randomized policy discussed in the next section.

The coordination ratio obtained may remind the reader of similar bounds for weighted congestion games [3]. It is important to stress that our bounds do not follow from these results. What *can* be deduced by applying the arguments of Azar et al. [3] to our setting is that $C^{PS}(\boldsymbol{x})/C^{PS}(\boldsymbol{x}^*) \leq \phi + 1$ for any Nash assignment $\boldsymbol{x}$ for the very restricted set of instances in which every pair of jobs j, j' satisfy $\rho_{ij} = \rho_{ij'}$ on each machine i, or in other words, all jobs scheduled on the same machine face the same completion time. Our result shows that, for arbitrary ρ_{ij} values, the ratio does not get any worse even when we compare against the stronger benchmark of $C^{SR}(\boldsymbol{x}^*)$.

In the case of equal weights, we obtain a slightly improved bound using the following lemma instead of the Cauchy-Schwartz inequality. This is a tighter version of an inequality initially used by Christodoulou and Koutsoupias [18]:

Lemma 4.5. *For every pair of nonnegative integers k and k^*,*

$$k^*(k+1) \leq \frac{1}{3}k^2 + \frac{5}{6}k^*(k^*+1).$$

Proof. This translates to showing that for all nonnegative integers k and k^*,

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

Rewriting the left hand side as $2(k - \frac{3}{2}k^*)^2 + \frac{1}{2}k^{*2} - k^*$, we see immediately that the inequality holds for $k^* \geq 2$ and $k^* = 0$. In the case $k^* = 1$, the required inequality simplifies to $k^2 - 3k + 2 \geq 0$ which is true for all integral k. □

Theorem 4.6. *The price of anarchy of EqualSharing for unrelated machines is at most 2.5. This bound is tight even for the restricted related machines model.*

Proof. Let $\boldsymbol{x}$ be any some assignment, and let $\boldsymbol{f} = \varphi(\boldsymbol{x})$. Since $w_j = 1$ for all j,

$$f_i(y) = |\{j \in J : x_j = i \text{ and } p_{ij} \geq y\}|,$$

and also

$$\begin{aligned}\eta(\boldsymbol{x}) &= \sum_i \sum_{j \in X_i} p_{ij} \\ &= \sum_{i \in I} \int_0^\infty \sum_{j \in X_i} \mathbf{1}_{y \le p_{ij}} dy \\ &= \sum_{i \in I} \int_0^\infty f_i(y) dy.\end{aligned}$$

For the unweighted case, just like ProportionalSharing reduces to EqualSharing, SmithRule reduces to ShortestFirst. Adapting Corollary 4.3 and Lemma 3.1 to these unweighted counterparts, we get

$$C^{ES}(\boldsymbol{x}) = \int_0^\infty f_i^2(y) dy \qquad \text{and} \qquad C^{SF}(\boldsymbol{x}) = \tfrac{1}{2} \int_0^\infty f_i(y)(f_i(y)+1) dy. \tag{5}$$

Now suppose $\boldsymbol{x}$ is a Nash equilibrium; take $\boldsymbol{x}^*$ to be any assignment, with $\boldsymbol{f}^* = \varphi(\boldsymbol{x}^*)$. We continue from (4):

$$\begin{aligned}C^{ES}(\boldsymbol{x}) &\le \langle \boldsymbol{f}, \boldsymbol{f}^* \rangle + \eta(\boldsymbol{x}^*) \\ &= \int_0^\infty f_i(y)(f_i^*(y)+1) dy \\ &\le \int_0^\infty \tfrac{1}{3} f_i^2(y)) + \tfrac{5}{6} f_i^*(y)(f_i^*(y)+1) dy \qquad \text{by Lemma 4.5} \\ &= \tfrac{1}{3} C^{ES}(\boldsymbol{x}) + \tfrac{5}{3} C^{SF}(\boldsymbol{x}^*) \qquad \text{by (5)}.\end{aligned}$$

This gives a price of anarchy bound of 2.5.

The tightness of the bound follows from Theorem 3 of [13]. The authors present a load balancing game lower bound, which is equivalent to assuming that all jobs have unit processing times and the machines are using EqualSharing; thus the same proof yields a (pure) price of anarchy lower bound for restricted related machines and unweighted jobs. □

Once again, all of the upper bounds also hold for the robust price of anarchy. On the negative side, we have the following (the proof of which can be found in Appendix A). Recall that a coordination mechanism is prompt if on any machine, the completion time of all jobs assigned to the machine are never larger than the sum of processing times of jobs on the machine. Equivalently, each machine uses its full capacity and does not delay the release of a job after its completion.

Proposition 4.7. *When jobs are anonymous, the coordination ratio of any deterministic prompt coordination mechanism is at least* 13/6.

4.2 Randomized Coordination Mechanism

In this section we examine the power of randomization and present Rand, a randomized policy that satisfies the following property: if two jobs j and j' are assigned to machine i, then

$$\mathbb{P}\{j \text{ precedes } j' \text{ in the ordering}\} = \frac{\rho_{ij'}}{\rho_{ij} + \rho_{ij'}}. \tag{6}$$

Recall $\rho_{ij} = p_{ij}/w_j$. A distribution over orderings with this property can be constructed as follows. Starting from the set of jobs X_i assigned to machine $i \in I$, select job $j \in X_i$ with probability

$\rho_{ij}/\sum_{k\in X_i}\rho_{ik}$, and schedule j at the end. Then remove j from the list of jobs, and repeat this process. Note that this policy is different from a simple randomized policy that orders jobs uniformly at random. In fact, this simpler policy is known to give an $\Omega(m)$ price of anarchy bound for the makespan objective [43], and the same family of examples developed in [43] gives an $\Omega(m)$ lower bound for this policy in our setting.

As we show below, this randomized policy outperforms any deterministic strongly local policy that has the "prompt" property defined above. In an attempt to explain this success, it is straightforward to verify that, unlike ProportionalSharing, this policy produces Pareto efficient schedules. One can actually show that it Pareto dominates ProportionalSharing. Yet, contrary to SmithRule, for any pair of jobs assigned to the same machine, there is positive probability that any one of the two is scheduled later, thus suffering a delay because of the other. In this sense, Rand gives high priority jobs the incentive to avoid crowded machines if they have better alternatives, but it does so without introducing very long delays.

Theorem 4.8. *The price of anarchy when using the Rand policy is at most* $32/15 = 2.133\cdots$. *Moreover, if the sum of the processing times of the jobs is negligible compared to the social cost of the optimal solution—more precisely,* $\eta(\boldsymbol{x}^*) = o(C^{SR}(\boldsymbol{x}^*))$*—this bound improves to* $\pi/2 + o(1)$*, which is tight.*

The high level approach for obtaining these upper bounds is in exactly the same spirit as the previous section: find an appropriate mapping φ from an assignment into a convenient inner product space. To make the mapping and inner product space easier to describe, we assume in this section that the processing times have been scaled so that the ratios ρ_{ij} are all integral. We also take κ large enough so that, except for infinite processing times, $\rho_{ij} \le \kappa$ for all $i \in I, j \in J$. These assumptions are inessential and easily removed.

An inner product space. The map φ we use gives the *signature* for each machine: in the unweighted case, this simply describes how many jobs of each size are assigned to the machine.

Definition 4.9. Given an assignment $\boldsymbol{x}$, its *signature* $\varphi(\boldsymbol{x}) \in \mathbb{R}_+^{m\times\kappa}$ is a vector indexed by a machine i and a processing time over weight ratio r; we denote this component by $\varphi(\boldsymbol{x})^i_r$. Its value is then defined as

$$\varphi(\boldsymbol{x})^i_r := \sum_{\substack{j\in X_i\\ \rho_{ij}=r}} w_j.$$

We also let $\varphi(\boldsymbol{x})^i$ denote the vector $(\varphi(\boldsymbol{x})^i_0, \varphi(\boldsymbol{x})^i_1, \ldots, \varphi(\boldsymbol{x})^i_\kappa)$.

Let M be the $\kappa\times\kappa$ matrix given by

$$M_{rs} = \frac{rs}{r+s}.$$

Lemma 4.10. *Let* $\boldsymbol{x}$ *be some assignment, and let* $\boldsymbol{u} = \varphi(\boldsymbol{x})$*. If job* j *is assigned to machine* i*, its expected completion time is given by*

$$c^R_j(\boldsymbol{x}) = (M\boldsymbol{u}^i)_{\rho_{ij}} + \tfrac{1}{2}p_{ij}.$$

If j *is not assigned to* i*, then its expected completion time upon switching to* i *would be*

$$c^R_j(\boldsymbol{x}) = (M\boldsymbol{u}^i)_{\rho_{ij}} + p_{ij}.$$

Proof. We consider case (i); (ii) is similar. So $x_j = i$. The expected completion time of job j on machine i is

$$\begin{aligned} c_j^R(\boldsymbol{x}) &= \sum_{k \in X_i \setminus \{j\}} p_{ik} \mathbb{P}\{\text{job } k \text{ ahead of job } j\} + p_{ij} \\ &= \sum_{k \in X_i \setminus \{j\}} p_{ik} \frac{\rho_{ij}}{\rho_{ij} + \rho_{ik}} + p_{ij} \\ &= \sum_{k \in X_i} p_{ik} \frac{\rho_{ij}}{\rho_{ij} + \rho_{ik}} + \tfrac{1}{2} p_{ij}. \end{aligned}$$

We can rewrite this in terms of the signature as

$$c_j^R(\boldsymbol{x}) = \sum_s u_s^i M_{\rho_{ij} s} + \tfrac{1}{2} p_{ij} = (M \boldsymbol{u}^i)_{\rho_{ij}} + \tfrac{1}{2} p_{ij}. \qquad \square$$

A crucial observation is the following:

Lemma 4.11. *The matrix M is positive definite.*

Proof. Let D be the diagonal matrix with $D_{rr} = r$. Then we have $M = DHD$, where the $\kappa \times \kappa$ matrix H is given by $H_{rs} = \frac{1}{r+s}$. This is a submatrix of the infinite Hilbert matrix $\left(\frac{1}{r+s-1}\right)_{r,s \in \mathbb{N}}$. The Hilbert matrix has the property that it is *totally positive* [17], meaning that the determinant of any submatrix is positive. It follows that H is positive definite, and hence so is M. $\square$

Thus we may define an inner product by

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle_R := \sum_{i \in I} (\boldsymbol{u}^i)^T M \boldsymbol{v}^i, \tag{7}$$

with an associated norm $\|\cdot\|_R$. In addition, the total cost $\sum_j w_j c_j^R(\boldsymbol{x})$ of an assignment $\boldsymbol{x}$ may be written in the convenient form

$$\begin{aligned} C^R(\boldsymbol{x}) &= \|\varphi(\boldsymbol{x})\|_R^2 + \tfrac{1}{2} \sum_{j \in J} w_j p_{x_j j} \\ &= \|\varphi(\boldsymbol{x})\|_R^2 + \tfrac{1}{2} \eta(\boldsymbol{x}). \end{aligned}$$

Competitiveness of Rand on a single machine. An interesting extra complication that occurs with this policy is that, unlike with ProportionalSharing, the inner product describing the cost of Rand is quite different to the one describing SmithRule. Since we ultimately need to compare against $C^{SR}(\boldsymbol{x}^*)$, we need to relate the cost of Rand and SmithRule. For this reason, the performance of Rand on even a *single* machine, compared to SmithRule, plays an important role.

So suppose we have n jobs with processing times p_j and weight w_j, for $j \le n$. The signature $\boldsymbol{u}$ is given by just $u_r = \sum_{j : p_j / w_j = r} w_j$. By considering (1), it follows that the weighted sum of completion times according to SmithRule is

$$\boldsymbol{u}^T S \boldsymbol{u} + \frac{1}{2} \sum_j w_j p_j,$$

where $S_{rs} = \frac{1}{2}\min\{r, s\}$. Compare this to the corresponding formula for Rand:

$$\boldsymbol{u}^T M \boldsymbol{u} + \frac{1}{2}\sum_j w_j p_j.$$

The extra $\sum_j w_j p_j$ terms only help, and in fact turn out to be negligible in the worst case example; ignoring them, the goal is to determine $\sup_{\boldsymbol{u}\geq \mathbf{0}} \frac{\boldsymbol{u}^T M \boldsymbol{u}}{\boldsymbol{u}^T S \boldsymbol{u}}$. So the question is closely related to the worst-case distortion between two norms (not quite, because of the nonnegativity constraint).

Interestingly, it turns out that this problem has been considered, and solved, in a different context. In [20], Chung, Hajela and Seymour consider the problem of *self-organizing sequential search*. In order to prove a tight bound on the performance of the "move-to-front" heuristic compared to the optimal ordering, they show:

Theorem 4.12 ([20])**.** *For any sequence $u_1, u_2, \ldots, u_k$ with $u_r > 0$ for all r,*

$$\sum_{r,s} u_r u_s \frac{rs}{r+s} < \frac{\pi}{4}\sum_{r,s} u_r u_s \min\{r, s\}.$$

(We also present a quite different proof of the theorem in Appendix B.) Thus on a single machine, Rand costs at most a factor $\frac{\pi}{2}$ more than SmithRule. Moreover, this is tight [35] (take $p_j = 1/j^2$, $w_j = 1$, and let $n \to \infty$). Of course, it follows immediately that for any number of machines and any assignment $\boldsymbol{x}$,

$$C^R(\boldsymbol{x}) \leq \frac{\pi}{2} C^{SR}(\boldsymbol{x}). \tag{8}$$

All in all, we find that $\pi/2$ is a tight upper bound on the competitiveness of Rand on a single machine. The following lemma (which may also be cast as a norm distortion question), is much more easily demonstrated:

Lemma 4.13. *For any assignment $\boldsymbol{x}$, we have $C^R(\boldsymbol{x}) \leq 2C^{SR}(\boldsymbol{x}) - \eta(\boldsymbol{x})$.*

Proof. Consider a particular machine i. We have

$$\begin{aligned}
\sum_{j,k\in X_i} w_j w_k \frac{\rho_{ij}\rho_{ik}}{\rho_{ij}+\rho_{ik}} &= \sum_{j\neq k\in X_i} w_j w_k \frac{\rho_{ij}\rho_{ik}}{\rho_{ij}+\rho_{ik}} + \tfrac{1}{2}\sum_{j\in X_i} w_j p_{ij} \\
&\leq \sum_{j\neq k\in X_i} w_j w_k \min\{\rho_{ij}, \rho_{ik}\} + \tfrac{1}{2}\sum_{j\in X_i} w_j p_{ij} \\
&= \sum_{j,k\in X_i} w_j w_k \min\{\rho_{ij}, \rho_{ik}\} - \tfrac{1}{2}\sum_{j\in X_i} w_j p_{ij}.
\end{aligned}$$

Summing over all machines gives

$$C^R(\boldsymbol{x}) - \tfrac{1}{2}\eta(\boldsymbol{x}) \leq 2\big(C^{SR}(\boldsymbol{x}) - \tfrac{1}{2}\eta(\boldsymbol{x})\big) - \tfrac{1}{2}\eta(\boldsymbol{x})$$

from which the bound is immediate. □

The upper bound. We are now ready to prove the main theorem of this section.

Proof of Theorem 4.8. Let $\boldsymbol{x}$ be the assignment at a Nash equilibrium, and $\boldsymbol{x}^*$ be any arbitrary assignment, and let $\boldsymbol{u} = \varphi(\boldsymbol{x})$ and $\boldsymbol{u}^* = \varphi(\boldsymbol{x}^*)$.

From the Nash condition and Lemma 4.10, we obtain

$$\begin{aligned} C^R(\boldsymbol{x}) &\leq \sum_{j\in J} w_j c_j^R(\boldsymbol{x}_{-j}, x_j^*) \\ &\leq \sum_{i\in I}\sum_{j\in X_i^*} w_j M(\boldsymbol{u}^i)_{\rho_{ij}} \ + \ \eta(\boldsymbol{x}^*) \\ &= \sum_{i\in I} (\boldsymbol{u}^{*i})^T M \boldsymbol{u}^i \ + \ \eta(\boldsymbol{x}^*) \\ &= \langle \boldsymbol{u}^*, \boldsymbol{u}\rangle_R + \eta(\boldsymbol{x}^*). \end{aligned}$$

Applying Cauchy-Schwartz,

$$\begin{aligned} C^R(\boldsymbol{x}) &\leq \|\boldsymbol{u}^*\|_R \|\boldsymbol{u}\|_R + \eta(\boldsymbol{x}^*) \\ &\leq \tfrac{2}{3}\|\boldsymbol{u}^*\|_R^2 + \tfrac{3}{8}\|\boldsymbol{u}\|_R^2 + \eta(\boldsymbol{x}^*), \end{aligned} \tag{9}$$

Now recalling the definition of φ and applying Lemma 4.13, we obtain

$$\begin{aligned} C^R(\boldsymbol{x}) &\leq \tfrac{2}{3}(C^R(\boldsymbol{x}^*) - \tfrac{1}{2}\eta(\boldsymbol{x}^*)) + \tfrac{3}{8}(C^R(\boldsymbol{x}) - \tfrac{1}{2}\eta(\boldsymbol{x})) + \eta(\boldsymbol{x}^*) \\ &\leq \tfrac{2}{3}(2C^{SR}(\boldsymbol{x}^*) - \tfrac{3}{2}\eta(\boldsymbol{x}^*)) + \tfrac{3}{8}(C^R(\boldsymbol{x}) - \tfrac{1}{2}\eta(\boldsymbol{x})) + \eta(\boldsymbol{x}^*) \\ &\leq \tfrac{4}{3}C^{SR}(\boldsymbol{x}^*) + \tfrac{3}{8}C^R(\boldsymbol{x}). \end{aligned}$$

This gives a coordination ratio of 32/15.

Suppose now that $\eta(\boldsymbol{x}^*)$ is very small; $\eta(\boldsymbol{x}^*) \leq \epsilon C^{SR}(\boldsymbol{x}^*)$ for some $\epsilon > 0$. Then we may continue from (9):

$$\begin{aligned} C^R(\boldsymbol{x}) &\leq \|\boldsymbol{u}^*\|_R \|\boldsymbol{u}\|_R + \epsilon C^{SR}(\boldsymbol{x}^*) \\ &\leq \sqrt{C^R(\boldsymbol{x}^*)\cdot C^R(\boldsymbol{x})} + \epsilon\sqrt{C^R(\boldsymbol{x}^*)\cdot C^R(\boldsymbol{x})}. \end{aligned}$$

Thus

$$C^R(\boldsymbol{x})/C^R(\boldsymbol{x}^*) \leq (1+\epsilon)^2.$$

So if $\eta(\boldsymbol{x}^*) = o(C^{SR}(\boldsymbol{x}))$, we obtain from (8) that $C^R(\boldsymbol{x})/C^{SR}(\boldsymbol{x}^*) \leq \pi/2 + o(1)$. □

As noted in Appendix A, a slight modification of the construction used to prove Proposition 4.7 can be used to show that the worst-case price of anarchy of Rand is at least 5/3.

5 Potential Games and an Algorithmic Application

Potential games. Under SmithRule it may happen that no pure Nash equilibrium exists [24]. Here we show that ProportionalSharing and Rand both induce exact potential games, which hence always have pure Nash equilibria. It also follows from this that certain natural best response dynamics very quickly converge to solutions which cost at most slightly larger than the price of anarchy bound; we discuss this in more detail at the end of this section.

The following theorem generalizes [27, Theorem 3], which addresses EqualSharing (i.e., the unweighted case).

Theorem 5.1. *The* ProportionalSharing *mechanism induces exact potential games, with potential*

$$\Phi^{PS}(\boldsymbol{x}) = \tfrac{1}{2}C^{PS}(\boldsymbol{x}) + \tfrac{1}{2}\eta(\boldsymbol{x}). \tag{10}$$

Likewise, the Rand *mechanism yields exact potential games with potential*

$$\Phi^{R}(\boldsymbol{x}) = \tfrac{1}{2}C^{R}(\boldsymbol{x}) + \tfrac{1}{2}\eta(\boldsymbol{x}). \tag{11}$$

Proof. We give the proof for ProportionalSharing; the proof for Rand is similar.

Consider an assignment $\boldsymbol{x}$ and a job $j \in J$, and let i be the machine to which j is assigned. Define $\boldsymbol{x'}$ as the assignment differing from $\boldsymbol{x}$ only in that job j moves to some machine $i' \neq i$.

We may write the change in the potential function as

$$\Phi^{PS}(\boldsymbol{x'}) - \Phi^{PS}(\boldsymbol{x}) = \sum_{k \in J} D_k \; + \; \tfrac{1}{2}w_j(p_{i'j} - p_{ij}), \tag{12}$$

where

$$D_k = \tfrac{1}{2}w_k \left(c_k^{PS}(\boldsymbol{x'}) - c_k^{PS}(\boldsymbol{x})\right).$$

Consider a job $k \neq j$ on machine i. Since only job j left the machine, we have from Lemma 4.1 that

$$c_k^{PS}(\boldsymbol{x'}) - c_k^{PS}(\boldsymbol{x}) = -w_j \min\{\rho_{ij}, \rho_{ik}\}.$$

Thus

$$\begin{aligned}\sum_{k \in X_i \setminus \{j\}} D_k &= -\tfrac{1}{2}w_j \sum_{k \in X_i \setminus \{j\}} w_k \min\{\rho_{ij}, \rho_{ik}\} \\ &= -\tfrac{1}{2}w_j(c_j^{PS}(\boldsymbol{x}) + p_{ij}).\end{aligned}$$

Similarly, considering jobs on i' yields

$$\begin{aligned}\sum_{k \in X_{i'}} D_k &= \tfrac{1}{2}w_j \sum_{k \in X_{i'}} w_k \min\{\rho_{i'j}, \rho_{i'k}\} \\ &= \tfrac{1}{2}w_j(c_j^{PS}(\boldsymbol{x'}) - p_{i'j}).\end{aligned}$$

All other jobs are unaffected by the change, and so do not contribute to (12). Summing all terms (including D_j), we obtain

$$\Phi^{PS}(\boldsymbol{x'}) - \Phi^{PS}(\boldsymbol{x}) = w_j(c_j^{PS}(\boldsymbol{x'}) - c_j^{PS}(\boldsymbol{x})),$$

exactly the change in the cost of job j. □

A combinatorial approximation algorithm. Minimizing the *unweighted* sum of completion times is polynomial time solvable, even for unrelated machines [41, 11]. For identical parallel machines, the ShortestFirst policy leads to an optimal schedule at any pure Nash equilibrium [23, 43]. On the other hand, minimizing the weighted sum of completion times is NP-complete even for identical machines [47]. This special case does admit a *polynomial time approximation scheme* (PTAS) however: for any $\epsilon > 0$, a solution only a factor $1 + \epsilon$ more expensive than the optimal one can be found in polynomial time [58]. Recall that the cost of the optimal solution, which we will denote by OPT, is simply $C^{SR}(\boldsymbol{x^*})$.

By contrast, the general case of unrelated machines is APX-hard [40]—no PTAS is possible. A sequence of papers gave improving constant-factor approximation algorithms, all based on rounding a linear or convex programming relaxation. The first was a 16/3-approximation algorithm [36], based on rounding an appropriate linear programming relaxation. An improvement to $\frac{3}{2}+\epsilon$, again based on linear programming, was given in [53]. Finally the best currently known factor, a $\frac{3}{2}$-approximation, was obtained based on a convex quadratic relaxation of the problem [55, 57].

In this section, we will give a very simple and *combinatorial* approximation algorithm. While it does not quite match the best factor of $\frac{3}{2}$, it achieves a factor of $2+\epsilon$, for any $\epsilon > 0$.

The basic idea is as follows. If we could compute a Nash equilibrium of the game induced by a policy with a coordination ratio of γ, this Nash equilibrium schedule would have a social cost at most γ times the optimum. The algorithm computing this Nash equilibrium would therefore be a γ approximation algorithm for the optimization problem. Of course, there is no longer any need to keep to the suboptimal scheduling that any policy apart from SmithRule would yield. Once we have the Nash assignment $\boldsymbol{x}$, we can switch to using SmithRule, as this step will always decrease the social cost. In what follows we carefully choose a policy that has a small coordination ratio, but at the same time guarantees that the cost will decrease by half after switching to SmithRule. This way, we can guarantee an approximation factor that is better than the best price of anarchy bound that we managed to achieve.

The policy we use, which we call Approx, is a variation of ProportionalSharing with some additional delays. Schedule the jobs exactly as in ProportionalSharing, but hold each job j back by an additional duration equal to its processing time. In other words, the completion time of any job j under an assignment $\boldsymbol{x}$ is

$$c_j^A(\boldsymbol{x}) = c_j^{PS}(\boldsymbol{x}) + p_{x_j j}.$$

Comparing Lemma 3.1 and Corollary 4.3, we see that

$$C^{PS}(\boldsymbol{x}) = 2C^{SR}(\boldsymbol{x}) - \eta(\boldsymbol{x}).$$

Thus

$$C^A(\boldsymbol{x}) = C^{PS}(\boldsymbol{x}) + \eta(\boldsymbol{x}) = 2C^{SR}(\boldsymbol{x}).$$

This will give us a saving of a factor of 2 when we switch from using the Approx policy to SmithRule. It turns out that Approx has a coordination ratio of 4; thus for any Nash equilibrium $\boldsymbol{x}$ with respect to this policy, $C^{SR}(\boldsymbol{x}) \leq 2OPT$.

Unfortunately we do not know how to compute an equilibrium allocation to this game (similarly for ProportionalSharing and Rand, in fact). Despite this, we will show that a natural best response dynamics will converge in polynomial time to some assignment of cost at most $(2+\epsilon)OPT$ for any $\epsilon > 0$. This will follow from general results on the robust price of anarchy proved by Roughgarden [51], drawing on work by Awerbuch et al. [4] and Chien and Sinclar [16]. We will actually prove everything we need, primarily in order to be able to give a precise stopping condition for our algorithm, something which is not quite explicit in [51]. It also demonstrates that there is no difficulty in extending the proofs to price of anarchy bounds on coordination mechanisms rather than games, although this is fairly immediate from a consideration of the original proofs.

Consider the following natural best response dynamics: simply pick the job which can improve its disutility (weighted completion time) the most by deviating, and allow that job to move. Following [51] we will call this the *maximum-gain* best response dynamics. Given some coordination mechanism α and configuration $\boldsymbol{x}$, let

$$\Delta_j^\alpha = w_j \left(c_j^\alpha(\boldsymbol{x}) - c_j^\alpha(\boldsymbol{x}_{-j}, x'_j) \right) \qquad \text{for any } j \in J,$$

where x'_j is the best responce move for player j. Also let

$$\Delta^\alpha(\boldsymbol{x}) = \sum_{j \in J} \Delta^\alpha_j.$$

Definition 5.2. An assignment $\boldsymbol{x}$ is an *ϵ-equilibrium* if $\Delta^\alpha(\boldsymbol{x}) < \epsilon C^\alpha(\boldsymbol{x})$.

The full algorithm using these dynamics is described in Algorithm 1. Note that this local search algorithm is using a non-trivial set of local moves implicitly defined by the Approx coordination mechanism. The choice of the coordination mechanism itself is rather counter-intuitive and based on the game theoretic intuition explained in previous sections. It therefore seems unlikely that such an algorithm could be designed without the game theoretic perspective.

Algorithm 1: A factor $2+\epsilon$ approximation algorithm for minimizing $\sum_{j \in J} w_j c^{SR}_j(\boldsymbol{x})$.

1 Assign each job to a machine on which it has minimum processing time.
2 Using the Approx policy, run basic dynamics until an $\epsilon/4$-equilibrium $\boldsymbol{x}$ is obtained.
3 Return assignment $\boldsymbol{x}$, scheduled according to SmithRule.

In order to bound the running time of our local-search algorithm we use the following theorem, slightly adapted from [51, Proposition 2.6], which is in turn based on [4].

Proposition 5.3. *[51] Let α be a (λ, μ)-smooth coordination mechanism, let $\boldsymbol{x}^0$ be any initial configuration, and let $\hat{\boldsymbol{x}}$ be the global minimizer of Φ^α. Then for any $\epsilon > 0$, maximum-gain best response dynamics generates an ϵ-equilibrium $\boldsymbol{x}$ in at most $O\left(\frac{n}{\epsilon} \log\left(\frac{\Phi^\alpha(\boldsymbol{x}^0)}{\Phi^\alpha(\hat{\boldsymbol{x}})}\right)\right)$ steps, and this assignment satisfies $C^\alpha(\boldsymbol{x}) \leq \frac{\lambda}{1-\mu-\epsilon} OPT$.*

Proof. By the definition of (λ, μ)-smooth, we have

$$\sum_{j \in J} w_j c^\alpha_j(\boldsymbol{x}_{-j}, x^*_j) \leq \mu C^\alpha(\boldsymbol{x}) + \lambda OPT.$$

Thus $\Delta^\alpha(\boldsymbol{x}) \geq (1-\mu) C^\alpha(\boldsymbol{x}) - \lambda OPT$, and so if $\boldsymbol{x}$ is an ϵ-equilibrium,

$$\epsilon C^\alpha(\boldsymbol{x}) \geq \Delta^\alpha(\boldsymbol{x}) \geq (1-\mu) C^\alpha(\boldsymbol{x}) - \lambda OPT,$$

implying the required cost bound.

Let $\boldsymbol{x}^t$ be the assignment after t steps of basic dynamics. Suppose that $\boldsymbol{x}^t$ is not an ϵ-equilibrium, so $\Delta^\alpha(\boldsymbol{x}^t) > \epsilon C^\alpha(\boldsymbol{x}^t)$. Then if j is the player which can improve the most, we must have $\Delta^\alpha_j(\boldsymbol{x}^t) \geq \frac{\epsilon}{n} C^\alpha(\boldsymbol{x}^t)$. Then since $C^\alpha(\boldsymbol{x}^t) \leq \Phi^\alpha(\boldsymbol{x}^t)$ and $\Phi^\alpha(\boldsymbol{x}^{t+1}) = \Phi^\alpha(\boldsymbol{x}^t) - \Delta^\alpha_j(\boldsymbol{x}^t)$, we have

$$\Phi^\alpha(\boldsymbol{x}^{t+1}) \leq (1 - \tfrac{\epsilon}{n}) \Phi^\alpha(\boldsymbol{x}^t).$$

Thus if no ϵ-equilibrium is found in the first T steps,

$$\Phi^\alpha(\hat{\boldsymbol{x}}) \leq \Phi^\alpha(\boldsymbol{x}^T) \leq (1 - \tfrac{\epsilon}{n})^T \Phi^\alpha(\boldsymbol{x}^0).$$

This yields the required bound on the number of steps. □

We omit the proofs of the following two lemmas, which are essentially identical to those of Theorem 5.1 and Theorem 4.4 respectively:

Lemma 5.4. *The* Approx *coordination mechanism induces an exact potential game, with potential function*

$$\Phi^A(\boldsymbol{x}) = \tfrac{1}{2}C^A(\boldsymbol{x}) + \eta(\boldsymbol{x}).$$

Lemma 5.5. *The* Approx *coordination mechanism is* $(3, 1/4)$*-smooth.*

We may now prove the main result of this section.

Theorem 5.6. *For any* $0 < \epsilon < 1$*, Algorithm 1 runs in polynomial time, and returns a schedule of cost at most* $(2+\epsilon)OPT$*.*

Proof. We first argue that $\Phi^A(\boldsymbol{x}^0) \leq (n+1)\Phi^A(\hat{\boldsymbol{x}})$, where $\hat{\boldsymbol{x}}$ is a global minimizer of Φ^A. Consider two jobs j and k assigned to some machine i under $\boldsymbol{x}^0$, such that j is processed before k under Smith's rule. Then $\rho_{ij} \leq \rho_{ik}$. The total contribution to the cost $C^{SR}(\boldsymbol{x}^0)$ due to the delay of job k by job j is $w_k p_{ij}$. But we have

$$w_k p_{ij} = w_j w_k \rho_{ij} \leq \tfrac{1}{2}(w_j^2 + w_k^2)\rho_{ij} \leq \tfrac{1}{2}(w_j p_{ij} + w_k p_{ik}).$$

Summing over all pairs of jobs processed on the same machine, the total contribution to $C^{SR}(\boldsymbol{x}^0)$ due to delays is at most

$$\sum_{i\in I} \sum_{j\neq k\in X_i^0} \tfrac{1}{2}(w_j p_{ij} + w_k p_{ik}) \leq (n-1)\sum_{i\in I}\sum_{j\in X_i^0} w_j p_{ij} = (n-1)\eta(\boldsymbol{x}^0).$$

Hence $C^{SR}(\boldsymbol{x}^0) \leq n\eta(\boldsymbol{x}^0)$, and so

$$\begin{aligned}
\Phi^A(\boldsymbol{x}^0) &\leq C^{SR}(\boldsymbol{x}^0) + \eta(\boldsymbol{x}^0) \\
&\leq (n+1)\eta(\boldsymbol{x}^0) \\
&\leq (n+1)\eta(\hat{\boldsymbol{x}}) \qquad \text{(since } \boldsymbol{x}^0 \text{ minimizes } \eta\text{)} \\
&\leq (n+1)\Phi^A(\hat{\boldsymbol{x}}).
\end{aligned}$$

Since Approx is $(3, 1/4)$-smooth, by Proposition 5.3 the algorithm returns an assignment $\boldsymbol{x}$ of cost $C^A(\boldsymbol{x}) \leq \frac{3}{1-1/4-\epsilon/4}OPT$ in $O(\frac{n\log n}{\epsilon})$ steps of best response dynamics. Thus the algorithm runs in polynomial time, and simplifying, $C^{SR}(\boldsymbol{x}) \leq (2+\epsilon)OPT$. □

Since we also have robust price of anarchy bounds for ProportionalSharing and Rand, and these both induce exact potential games, fast convergence statements can be made for these policies as well. As well as the maximum-gain best response dynamics used in Algorithm 1, best response dynamics where a random player is chosen at each round also leads to convergence, with high probability [51].

6 Concluding remarks

On mapping machines to edges of a parallel link network, the machine scheduling problem for the case of related machines becomes a special case of general selfish routing games. In this context, the ordering policies on machines correspond to local queuing policies at the edges of the network. From this perspective, it would be interesting to generalize our results to network routing games. Designing such local queuing policies would be an important step toward more realistic models of selfish routing games when the routing happens over time [38, 29, 44, 22]. We hope that our new

technique along with the policies proposed in this paper could serve as a building block toward this challenging problem.

All the mechanisms discussed here are strongly local. For the case of the makespan objective, one can improve the coordination ratio from $\Theta(m)$ to $\Theta(\log m)$ by using local policies instead of just strongly local policies. It remains open whether there are local policies that perform even better than our strongly local ones. In particular, we do not know of any local policy that does better than Rand.

Acknowledgements. We thank Tanmoy Chakraborty and Andy Skrzypacz for helpful discussions and Ioannis Caragiannis for pointing out related literature. The fourth author also thanks Yossi Azar for initial discussions about the subject of study of this paper. Part of this work was done while the second and last authors were visiting EPFL; we thank Fritz Eisenbrand for his hospitality.

References

[1] F. Abed and C.-C. Huang. Preemptive coordination mechanisms for unrelated machines. In *ESA*, pages 12–23, 2012.

[2] 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.

[3] B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. In *STOC*, pages 57–66, 2005.

[4] B. Awerbuch, Y. Azar, A. Epstein, V.S. Mirrokni, and A. Skopalik. Fast convergence to nearly optimal solutions in potential games. In *ACM Conference on Electronic Commerce*, pages 264–273, 2008.

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

[6] Y. Azar, K. Jain, and V.S. Mirrokni. (Almost) optimal coordination mechanisms for unrelated machine scheduling. In *SODA*, pages 323–332, 2008.

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

[8] A. Bagchi. Stackelberg differential games in economic models. *Springer-Verlag*, 1984.

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

[10] A. Borodin, M.N. Nielsen, and C. Rackoff. (Incremental) priority algorithms. *Algorithmica*, 37:295–326, 2003. 10.1007/s00453-003-1036-3.

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

[12] I. Caragiannis. Efficient coordination mechanisms for unrelated machine scheduling. *Algorithmica*, pages 1–29, April 2012.

[13] I. Caragiannis, M. Flammini, C. Kaklamanis, P. Kanellopoulos, and L. Moscardelli. Tight bounds for selfish and greedy load balancing. *Algorithmica*, 61(3):606–637, 2011.

[14] I. Caragiannis, C. Kaklamanis, and P. Kanellopoulos. Taxes for linear atomic congestion games. *ACM Transactions on Algorithms*, 7(1):13, 2010.

[15] Y. Chevaleyre, P.E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J.A. Padget, S. Phelps, J.A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica (Slovenia)*, 30(1):3–31, 2006.

[16] S. Chien and A. Sinclair. Convergence to approximate Nash equilibria in congestion games. *Games and Economic Behavior*, 71(2):315–327, 2011.

[17] M.-D. Choi. Tricks or treats with the Hilbert matrix. *The American Mathematical Monthly*, 90(5):301–312, May 1983.

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

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

[20] F. Chung, D. Hajela, and P. D. Seymour. Self-organizing sequential search and Hilbert's inequalities. *Journal of Computer and System Sciences*, 36(2):148–157, April 1988.

[21] R. Cole, Y. Dodis, and T. Roughgarden. How much can taxes help selfish routing? *J. Comput. Syst. Sci.*, 72(3):444–467, 2006.

[22] R. Cominetti, J.R. Correa, and O. Larre. Existence and uniqueness of equilibria for flows over time. In *ICALP (part II)*, pages 552–563, 2011.

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

[24] J.R. Correa and M. Queyranne. Efficiency of equilibria in restricted uniform machine scheduling with total weighted completion time as social cost. *Naval Research Logistics*, 59(5):384–395, 2012.

[25] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. *ACM Trans. Algorithms*, 3(1):4:1–4:17, February 2007.

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

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

[28] E. Even-Dar, A. Kesselman, and Y. Mansour. Convergence time to Nash equilibrium in load balancing. *ACM Trans. Algorithms*, 3(3), August 2007.

[29] B. Farzad, N. Olver, and A. Vetta. A priority-based model of routing. *Chicago Journal of Theoretical Computer Science*, 2008(1).

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

[31] 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.

[32] L. Fleischer and Z. Svitkina. Preference-constrained oriented matching. In *ANALCO*, pages 56–65, 2010.

[33] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, nov 1998.

[34] M. Gairing, T. Lücking, M. Mavronicolas, and B. Monien. Computing nash equilibria for scheduling on restricted parallel links. *Theory Comput. Syst.*, 47(2):405–432, 2010.

[35] G.H. Gonnet, J.I. Munro, and H. Suwanda. Exegesis of self-organizing linear search. *SIAM Journal on Computing*, 10(3):613–637, 1981.

[36] 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.

[37] J.D. Hartline and T. Roughgarden. Optimal mechanism design and money burning. In *STOC*, pages 75–84, 2008.

[38] M. Hoefer, V.S. Mirrokni, H. Röglin, and S.-H. Teng. Competitive routing over time. *Theoretical Computer Science*, 412(39):5420–5432, 2011.

[39] R. Hoeksma and M. Uetz. The price of anarchy for minsum related machine scheduling. In *WAOA*, pages 261–273, 2011.

[40] H. Hoogeveen, P. Schuurman, and G.J. Woeginger. Non-approximability results for scheduling problems with minsum criteria. *INFORMS J. Comput.*, 13(2):157–168, 2001.

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

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

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

[44] R. Koch and M. Skutella. Nash equilibria and the price of anarchy for flows over time. *Theor. Comp. Sys.*, 49(1):71–97, July 2011.

[45] 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.

[46] E. Koutsoupias and C.H. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3(2):65–69, 2009.

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

[48] L.L. Peterson, T.E. Anderson, D.E. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the internet. *Computer Communication Review*, 33(1):59–64, 2003.

[49] M. Pinedo. *Scheduling: Theory, Algorithms and Systems*. Springer, third edition, 2008.

[50] T. Roughgarden. Stackelberg scheduling strategies. *SIAM Journal on Computing*, 33(2):332–350, 2004.

[51] T. Roughgarden. Intrinsic robustness of the price of anarchy. In *STOC*, pages 513–522, 2009. *Expository version published in C. ACM, July 2012.*

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

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

[54] P. Schuurman and T. Vredeveld. Performance guarantees of local search for multiprocessor scheduling. *INFORMS J. Comput.*, 19(1):52–63, 2007.

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

[56] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, December 2008.

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

[58] 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.

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

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

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

A Lower Bounds

Proof of Theorem 3.3. We begin by presenting the family of game instances that leads to a pure price of anarchy approaching 4 for games induced by SmithRule in the restricted identical machines model [24], and then show how to generalize the lower bound based on this construction.

There are m machines and k groups of unweighted unit-length jobs $g_1, \ldots, g_k$, where group g_r has m/r^2 jobs. We assume that m is such that all groups have integer size and let j_{rs} denote the s-th job of the r-th group. A job j_{rs} can be assigned to machines $1, \ldots, s$, and we assume that for two jobs j_{rs} and $j_{r's'}$ with $s < s'$, $j_{r's'}$ has higher priority than j_{rs} (if $s = s'$, the ordering can be arbitrary).

If every job j_{rs} is assigned to machine s, there are exactly m/r^2 jobs with completion time r ($1 \leq r \leq k$), which leads to a total cost of $m \sum_{r=1}^{k} 1/r$. On the other hand, consider the following assignment. Process the jobs in order of priority, highest to lowest. For each, consider the set of machines which minimizes its completion time (given the already assigned jobs), and assign it to

the machine in this set with smallest index. This gives a (pure) Nash equilibrium by construction, since the completion time at the point when a job is assigned is unaffected by the assignment of later jobs. In [24], it is shown that this assignment has total cost $\Omega(4m \sum_{r=1}^{k} 1/r)$. Figure 2 demonstrates the case $k = 2$.

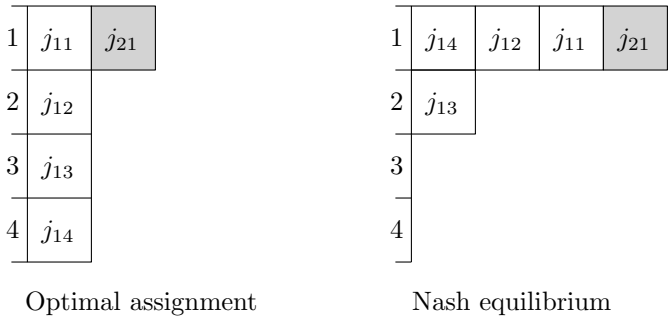

Figure 2: Lower bound instance for $m = 4$, $k = 2$, showing the optimal solution and a Nash equilibrium with high social cost.

Fixed ordering policies. We seek to reproduce the above construction in a setting in which each machine has its own ordering of the jobs. Let $\boldsymbol{x}$ be the Nash equilibrium described above, and $\boldsymbol{x}^*$ the optimal assignment. Note that $x_j \leq x_j^*$ for every job j. Since the job ordering under the optimal assignment does not affect the cost, we only need to make sure that for any two jobs j, j' with $x_{j'}^* < x_j^*$, j gets higher priority than j' on $x_{j'}^*$.

Given a specific lower bound instance for SmithRule, we have n job *slots*, each defined by the pair of machines x_j and x_j^*; the job assigned to this slot will be restricted to these two machines. Given a set of ordering policies, each machine has its own strictly ordered list of all n jobs. What we need to do is assign a specific job to each slot so that the ordering restrictions as specified in the previous paragraph comply with the lists. We start from the slot j with the greatest index x_j, and we assign the first job of machine x_j's list to this slot. We then erase this job from all lists and repeat. In case of a tie, that is if there is more than one slot with the same x_j, we first consider the slots with greater x_j^* machine index. This ensures that, given the Nash equilibrium assignment, if the job of some slot j deviates from machine x_j to machine x_j^*, it will suffer a cost at least as high as the one in the SmithRule instance; its cost in the Nash equilibrium, on the other hand, is the same as the one in the SmithRule instance. Therefore, $\boldsymbol{x}$ is a Nash equilibrium for this set of ordering policies. □

Proof of Proposition 4.7. The construction is a slight variant of one given in Caragiannis et al. [13] for load balancing games. We define the construction in terms of the *game graph*; a directed graph, with nodes corresponding to machines, and arcs corresponding to jobs. The interpretation of an arc (i^*, i) is that the corresponding machine is run on i at the Nash equilibrium, and i^* in the optimal solution (all jobs can only be run on at most two machines in the instance we construct).

Our graph consists of a binary tree of depth ℓ, with a path of length ℓ appended to each leaf of the tree. In addition, there is a loop at the endpoint of each path. All arcs are directed towards the root; the root is considered to be at depth zero. In the binary tree, on a machine at depth i, the processing time of any job that can run on that machine is $(3/2)^{\ell-i}$. In the chain, on a machine at distance k from the tree leaves all processing times are $(1/2)^k$.

By slightly perturbing the processing times of jobs on different machines, it is easily checked that if every job is run on the machine pointed to by its corresponding arc, the assignment is a pure

Nash equilibrium. The latter holds for arbitrary prompt strongly local coordination mechanisms so long as jobs are anonymous. On the other hand, if all jobs choose their alternative strategy, we obtain the optimal solution. A straightforward calculation shows that, in the limit $\ell \to \infty$, the ratio of the cost of the Nash equilibrium to the optimal cost converges to $13/6 > 2.166$. □

Rand. The previous instance can be easily modified to give a lower bound on the performance of Rand. Simply take the same instance but replace $3/2$ by $4/3$ and $1/2$ by $2/3$. The same assignment then gives a Nash equilibrium, and in this case the ratio of interest approaches $5/3$.

B The performance of Rand on a single machine

Proof of Theorem 4.12. We want to prove that for any sequence $u_1, \ldots, u_k$, $u_i \geq 0$, the following inequality holds:

$$\sum_i \sum_j u_i u_j \frac{ij}{i+j} \leq \frac{\pi}{4} \sum_i \sum_j u_i u_j \min\{i, j\}.$$

We will in fact prove that for any sequence $x_1, x_2, \ldots, x_n$, $x_i \in \mathbb{N}$,

$$\sum_i \sum_j \frac{x_i x_j}{x_i + x_j} < \frac{\pi}{4} \sum_i \sum_j \min\{x_i, x_j\}. \tag{13}$$

This implies the inequality in the statement, for the choice $u_r = |\{i : x_i = r\}|$, and hence clearly for any integer sequence (u_i). An obvious scaling argument then gives it for general nonnegative u_i.

Since both summations in (13) are symmetric, we may assume without loss of generality that $x_1 \geq \cdots \geq x_n \geq 0$. Then, we note that $\sum_{i=1}^n \sum_{j=1}^n \min\{x_i, x_j\} = 2\sum_{i=1}^n x_i(i - 1/2)$. Also, observe that the inequality is homogeneous so that proving the inequality is equivalent to proving that the optimal value of the following concave optimization problem is less than $\pi/2$:

$$z = \max \left\{ \sum_{i=1}^n \sum_{j=1}^n \frac{x_i x_j}{x_i + x_j} : \text{ s.t. } \sum_{i=1}^n x_i(i - 1/2) = 1,\ x_1 \geq \cdots \geq x_n \geq 0 \right\}.$$

Clearly $z \leq z'$, where

$$z' = \max \left\{ \sum_{i=1}^n \sum_{j=1}^n \frac{x_i x_j}{x_i + x_j} : \text{ s.t. } \sum_{i=1}^n x_i(i - 1/2) = 1,\ x_i \geq 0 \text{ for all } i = 1, \ldots, n \right\}.$$

Furthermore, we may assume that in an optimal solution all variables satisfy $x_i > 0$. Otherwise, we could consider the problem in a smaller dimension. Thus, the KKT optimality conditions state that for all $i = 1, \ldots, n$ we have

$$\mu(i - 1/2) = 2 \sum_{j=1}^n \left(\frac{x_j}{x_i + x_j} \right)^2. \tag{14}$$

Multiplying by x_i, summing over all i, and using $\sum_{i=1}^n x_i(i - 1/2) = 1$, we obtain:

$$\mu = 2 \sum_{i=1}^n \sum_{j=1}^n x_i \left(\frac{x_j}{x_i + x_j} \right)^2 = \sum_{i=1}^n \sum_{j=1}^n \frac{x_i x_j}{(x_i + x_j)^2} (x_i + x_j) = z'.$$

Now consider (14) with $i^* = \arg\max_i x_i(i-1/2)^2$. We have that

$$z' = \frac{2}{i^*-1/2}\sum_{j=1}^{n}\left(\frac{x_j}{x_{i^*}+x_j}\right)^2 \leq 2(i^*-1/2)^3\sum_{j=1}^{\infty}\left(\frac{1}{(i^*-1/2)^2+(j-1/2)^2}\right)^2.$$

Using standard complex analysis it can be shown that the latter summation equals

$$(\pi/2)\big((i^*-1/2)\pi\tanh(\pi(i^*-1/2))^2+\tanh(\pi(i^*-1/2))-\pi(i^*-1/2)\big),$$

which is less than $\pi/2$. □