Inner Product Spaces for MinSum Coordination Mechanisms[*]

Richard Cole [†]   José R. Correa [‡]   Vasilis Gkatzelis [§]   Vahab Mirrokni [¶]

Neil Olver [∥]

Abstract

We study policies aiming to minimize the weighted sum of completion times of jobs in the context of coordination mechanisms for selfish scheduling problems. Our goal is to design local policies that achieve a good price of anarchy in the resulting equilibria for unrelated machine scheduling. To obtain the approximation bounds, we introduce a new technique that while conceptually simple, seems to be quite powerful. The method entails mapping strategy vectors into a carefully chosen inner product space; costs are shown to correspond to the norm in this space, and the Nash condition also has a simple description. With this structure in place, we are able to prove a number of results, as follows.

First, we consider Smith's Rule, which orders the jobs on a machine in ascending processing time to weight ratio, and show that it achieves an approximation ratio of 4. We also demonstrate that this is the best possible for deterministic non-preemptive strongly local policies. Since Smith's Rule is always optimal for a given fixed assignment, this may seem unsurprising, but we then show that better approximation ratios can be obtained if either preemption or randomization is allowed.

We prove that ProportionalSharing, a preemptive strongly local policy, achieves an approximation ratio of 2.618 for the weighted sum of completion times, and an approximation ratio of 2.5 in the unweighted case. Again, we observe that these bounds are tight. Next, we consider Rand, a natural non-preemptive but *randomized* policy. We show that it achieves an approximation ratio of at most 2.13; moreover, if the sum of the weighted completion times is negligible compared to the cost of the optimal solution, this improves to $\pi/2$.

Finally, we show that both ProportionalSharing and Rand induce potential games, and thus always have a pure Nash equilibrium (unlike Smith's Rule). This also allows us to design the first *combinatorial* constant-factor approximation algorithm minimizing weighted completion time for unrelated machine scheduling. It achieves a factor of $2 + \epsilon$ for any $\epsilon > 0$, and involves imitating best response dynamics using a variant of ProportionalSharing as the policy.

1 Introduction

Traditionally, work in operations research has focused on finding globally optimal solutions for optimization problems. In tandem, computer scientists have long studied the effects of a lack of different kinds of resources, mainly the lack of computational resources in optimization. In designing massive decentralized systems, the *lack of coordination* among different participating agents has become an important consideration. This issue is typically addressed through distributed algorithms in which a central authority designs mechanisms (protocols) specifying the rules of the game, with the goal that the independent and selfish choices of the users result in a socially desirable outcome. To measure the performance of these algorithms, the global objective function (social cost) is evaluated at equilibrium points for selfish users. For games, probably the most accepted such measure is the *price of anarchy* [38], the worst case ratio of the social cost at a Nash equilibrium to that at a social optimum; the same measure can be used for coordination mechanisms; sometimes we call this their *approximation factor* to highlight that this is a distinct measure.

The by now standard approach to bound the price of anarchy (PoA) when social cost is taken to be the sum of individual costs works as follows [41]. First, the social cost is bounded by using the equilibrium conditions, noting that an individual is better off at equilibrium than she would be if she unilaterally changed her strategy to the one she would use in a centralized optimum. Second, the actual (weighted) sum of player costs is also upper bounded, using an appropriately chosen inequality, by a linear combination of the social cost of the equilibrium and the social cost of an optimal solution.

In this paper we establish a methodology to deal with the second step of this proof scheme. Our method interprets the sum in the second step as an inner product on a suitable space. Then, we apply the Cauchy-Schwartz inequality in the chosen inner product space, and go back to the original space by applying a minimum norm distortion inequality. Many of the existing results employ a special case of this approach in which the costs can be expressed in terms of quadratic polynomials to which the Cauchy-Schwartz inequality can be applied directly without the need for an intermediate inner product space. We apply our new method in the context of scheduling jobs on unrelated machines. Our method elucidates the hidden structure in the games we consider. Once the framework has been set up, our proofs become short and elegant, thus we anticipate that this method may prove useful elsewhere too.

Specifically, we consider the classic problem of scheduling n jobs on m unrelated machines from a game theoretic perspective. In this situation, job j takes time p_{ij} if processed on machine i, and also has an associated weight w_j. Although the central goal is to minimize the weighted sum of completion times of jobs, we consider the *scheduling game* in which each job is a fully informed player wanting to minimize its individual weighted completion time, while each machine announces a policy which it will follow in processing the jobs it is assigned. Our goal is to choose the policy so as to minimize the approximation ratio of the actual costs under this policy to the optimal costs obtainable under any policy. To this end, several approaches imposing incentives on self-interested agents have been proposed, including some using monetary transfers [7, 17, 26, 12], and others enforcing strategies on a fraction of users as a Stackelberg strategy [6, 37, 40, 49]. Ultimately one could also apply a VCG mechanism to achieve social efficiency. The main drawback of these methods is the need for global knowledge of the system. A different approach, and the focus of our paper, uses coordination mechanisms [15], which only require local computations.

More formally, a coordination mechanism [15, 35, 4, 10, 22] is a set of *local policies*, one per machine, specifying how the jobs assigned to that machine are scheduled. Here, *local* means that a machine's schedule must be a function only of the jobs it is assigned, allowing the policy to be implemented in a distributed fashion. We actually study *strongly* local policies, meaning that the

schedule of any machine i is a function only of the processing times p_{ij}, weights w_j and IDs of jobs assigned to it. It will also be useful (especially when considering lowerbounds) for us to restrict attention to policies that always use the full capacity of a machine, and release jobs immediately upon completion. We call such policies *prompt*.

Several local policies have been studied for machine scheduling problems in the context of both greedy and local search algorithms [34, 25, 42, 21, 1, 5, 8, 50], as well as coordination mechanisms [38, 20, 15, 35, 4, 10, 22]. Previous work mainly considered the makespan social cost as opposed to the weighted sum of completion times addressed here.

Our Results. Employing our new technique, we develop the first constant-factor approximate coordination mechanisms for the selfish machine scheduling problem for unrelated machines. We start by studying Smith's Rule [48], in which machines process jobs in increasing order of their processing time to weight ratio. Here the space that appropriately fits our method is L^2 and a norm distortion inequality is in fact not needed. We prove that the approximation factor for this policy is exactly 4, improving upon a result by Correa and Queyranne [19]. We also show that this is the best possible among all deterministic and non-preemptive strongly local coordination mechanisms, assuming the prompt property.

The constant approximation ratio for the weighted sum of completion times is in sharp contrast to the known super-constant inapproximability results for coordination mechanisms for the makespan function [4, 27] (e.g., an $\Omega(m)$ lower bound for the shortest-first coordination mechanism). In fact, it is still open whether there is a coordination mechanism with a constant approximation ratio for the makespan function.

Next, we go beyond the approximation ratio of 4 using preemptive[1] and randomized mechanisms. First, we consider a preemptive policy, generalizing that of Dürr and Thang [22], in which each machine splits its processing capacity among its assigned jobs in proportion to their weights. 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 approximation factor of 2.618. On the other hand, we prove that with anonymous jobs, no set of deterministic prompt policies, be they preemptive or not, can achieve a factor better than 2.166. To break this new barrier we consider a policy in which jobs are randomly, but non-uniformly, ordered, based on their processing time to weight ratio. Under this policy the appropriate space has to be carefully chosen and uses a rather nonstandard inner product, induced by a Hilbert matrix, whose i, j entry equals $1/(i+j-1)$. A norm distortion inequality is then needed to relate the norm in this space to the original cost of an optimal schedule, leading to yet another improvement in the approximation factor to 2.134. Moreover, we show a lower bound of $5/3 > 1.666$ for this policy.

Finally, inspired by our preemptive mechanism, along with the β-nice notion of [3], we design a new *combinatorial* $(2 + \epsilon)$-approximation algorithm for optimizing the weighted sum of completion times on unrelated machines. This improves on the approximation factor of our mechanisms and complements the known non-combinatorial constant-factor approximation algorithms: a linear programming based 16/3-approximation algorithm [30], then an improvement to $\frac{3}{2} + \epsilon$ again based on linear programming [43], and finally the best currently known factor, a $\frac{3}{2}$-approximation based on a convex quadratic relaxation [45, 46].

We obtain a number of other results, most of which are discussed in the appendices. For the unit weight case, using Smith's Rule, we obtain a constant upper bound on the price of anarchy by a reduction from the priority routing model of Farzad et al. [24], as shown in Appendix C; however, the resulting bound is not optimal. In the unit weight case, our preemptive mechanism simplifies

---

[1]By preemption we mean that the computation of a job is suspended and, implicitly, resumed later.

to one, called EqualSharing [22], in which jobs share the processing capacity of a machine equally. Then the approximation ratio is 2.5, which follows either by a careful analysis based on local moves, or using our method with a modified Cauchy-Schwartz inequality. In addition, in the case where the weighted sum of processing times is negligible compared to the total cost, our randomized policy has an approximation ratio of $\pi/2$, which is tight. The latter follows by an interesting norm distortion inequality obtained by Chung et al. [16], for which we provide an alternative shorter proof. Furthermore, although for the Smith's Rule policy pure equilibria may not exist [19], we show that all our preemptive and randomized mechanisms result in exact potential games. This implies that best-response dynamics of players converge to pure Nash Equilibria (PNE) and verifies that PNE always exist. While we present our results for pure strategies and pure Nash equilibria, we observe that all the results can be stated within the smoothness framework of Roughgarden [41], and so all the bounds hold for more general equilibrium concepts including mixed Nash equilibria and correlated equilibria. We assume that jobs aim at minimizing their weighted completion time in the case of deterministic policies, and expected weighted completion time for randomized ones.

It is important to stress that these bounds are on the price of anarchy (or approximation ratio) of *coordination mechanisms* and not that of games; thus these results do not follow from seemingly similar bounds for selfish routing [2]. The fact that our preemptive policy performs better than non-preemptive ones is in contrast to existing results for the makespan social cost function where the EqualSharing policy achieves an approximation ratio of $\Theta(m)$ [22], no better than ShortestFirst, which schedules jobs in increasing order of their processing times (i.e., Smith's Rule in the unweighted case). In order to explain this counter-intuitive result, we show that both our preemptive policy and our randomized policy penalize each job with an extra charge beyond its cost under Smith's Rule that is exactly equal to the externality its scheduling causes.

Other Related Work.   Scheduling problems have long been studied from a centralized optimization perspective. We adopt the standard three filed notation $\alpha|\beta|\gamma$ [29]. The first parameter defines the machine model, the last specifies the objective function, while the second will not concern us in this paper.

Minimizing the sum of completion times is polynomial time solvable even for unrelated machines [33, 9]. For identical parallel machines ($P||\sum c_j$), the ShortestFirst policy leads to an optimal schedule at any pure Nash equilibrium[2] [18]. On the other hand, minimizing the weighted sum of completion times is NP-complete even for identical machines ($P||\sum w_j c_j$) [39]. Although the latter admits a PTAS [47], the general unrelated case ($R||\sum w_j c_j$) is APX-hard [32] and constant factor approximation algorithms have been proposed [30, 43, 45, 46].

Coordination mechanism design was introduced by Christodoulou, Koutsoupias and Nanavati [15]. They analyzed the LongestFirst policy w.r.t. the makespan for identical machines ($P||C_{\max}$) and also studied a selfish routing game. Immorlica et al. [35] study four coordination mechanisms for different machine scheduling problems and survey the results for these problems. They further study the speed of convergence to equilibria and the existence of PNE for the ShortestFirst and LongestFirst policies. Azar, Jain, and Mirrokni [4] showed that the ShortestFirst policy and in fact any strongly local ordering policy (defined in Section 2) does not achieve an approximation ratio better than $\Omega(m)$. Additionally, they presented a non-preemptive local policy that achieves an approximation ratio of $O(\log m)$ and a policy that induces potential games and gives an approximatition ratio of $O(\log^2 m)$. Caragiannis [10] showed an alternative $O(\log m)$-approximate coordination mechanism that minimizes makespan for unrelated machine scheduling and does lead to potential games. Fleischer and Svitkina [27] show a lower bound of $\Omega(\log m)$ for all local or-

---

[2]In [35] it is shown that these equilibria are exactly the solutions generated by the shortest-first greedy algorithm.

dering policies. It is still open whether there exists a coordination mechanism (even preemptive or randomized) achieving a constant approximation ratio for the makespan objective function.

More recently, Dürr and Thang proved that the EqualSharing policy results in potential games, and achieves a PoA of $\Theta(m)$ for $R||C_{\max}$. In the context of coordination mechanisms, an instance for which a preemptive policy has an advantage over non-preemptive ones was also shown by Caragiannis [10], who presented a local preemptive policy with approximation factor $O(\log m/\log\log m)$, beating the lower bound of $\Omega(\log m)$ for local ordering policies [27]. Correa and Queyranne [19] study the problem of minimizing the weighted sum of completion times, show that Smith's Rule may induce games that do not have PNE, and that the price of anarchy under this policy is 4 in a more restricted environment than that considered here.

Coordination mechanisms are related to local search algorithms. Starting from a solution, a local search algorithm iteratively moves to a neighbor solution which improves the global objective. This is based on a neighborhood relation that is defined on the set of solutions. The local improvement moves in the local search algorithm correspond to the best-response moves of users in the game defined by the coordination mechanism. The speed of convergence and the approximation factor of local search algorithms for scheduling problems have been studied mainly for the makespan objective function [21, 23, 25, 34, 42, 44, 50, 1, 5]. Our combinatorial approximation algorithm for the weighted sum of completion time is the first local search algorithm for $R||\sum w_i C_i$ and is different from the previously studied algorithms for the makespan objective.

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 let w_j denote its weight (or importance or impatience). Our goal is to minimize the weighted sum of the completion times of the jobs, i.e. $\sum_{j \in J} w_j c_j$, where c_j is the completion time of job j. An assignment of jobs to machines is represented by a vector $\boldsymbol{x}$, where x_j gives the machine to which job j is assigned.

The main scheduling model we study is *unrelated* machine scheduling ($R||\sum w_j c_j$) in which the p_{ij}'s are arbitrary. Another model is the *restricted related* machines model in which each machine i has a speed q_i and each job j has a processing requirement p_j: job j can be scheduled only on a subset T_j of the machines, with processing time $p_{ij} = p_j/q_i$ for $i \in T_j$, and $p_{ij} = \infty$ otherwise. The *restricted identical* machines model is the special case of the restricted related machines model where all machines have the same speed.

A *coordination mechanism* is a set of local policies, one for each machine, that determines how to schedule the jobs assigned to that machine. It thereby defines a game in which there are n agents (jobs) and each agent's strategy set is the set of machines I. Given an assignment $\boldsymbol{x}$, the disutility of job j is its weighted completion time $w_j c_j(\boldsymbol{x})$, as determined by the policy on the machine x_j. The goal of each job is to choose a strategy (i.e., a machine) that minimizes its disutility. A strategy profile $\boldsymbol{x}$ is a *Nash equilibrium* if no player has an incentive to change strategy. Our goal is to design coordination mechanisms which give such incentives to the players, that selfish behavior leads to equilibria with low social cost.

A game is a *potential game* if there exists a potential function over strategy profiles such that any player's deviation leads to a drop of the potential function if and only if its cost drops. A potential game is *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 PNE.

We define a machine's policy to be *prompt* if the machine uses its full capacity and does not delay the release of any of its completed jobs. We say that a policy satisfies the *independence of irrelevant alternatives* or *IIA* property if for any pair of jobs, their relative ordering is independent

of what other jobs are assigned to the machine. This property property appears as an axiom in voting theory, bargaining theory and logic. Notice that deterministic non-preemptive policies with the IIA property can be described simply by a fixed ordering of all jobs; jobs are scheduled according to this order. Thus we call such policies *ordering* policies.

Here and throughout the paper, we use the shorthand notation ρ_{ij} for the ratio p_{ij}/w_j. 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. The main policies we discuss are the following:

SmithRule [48]: Jobs on machine i are scheduled consecutively in increasing order of ρ_{ij}. In the unweighted case, this reduces to the ShortestFirst policy.

ProportionalSharing: Jobs are scheduled in parallel using time-multiplexing. 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.

Rand: This randomized policy has the property that for any two jobs j, j' assigned to machine i, the probability that job j is run before job j' is exactly $\frac{\rho_{ij'}}{\rho_{ij}+\rho_{ij'}}$. Thus larger (w.r.t. ρ_{ij}) jobs are more likely to appear later in the ordering. We show how to implement this policy in Section 4.2.

For any configuration $\boldsymbol{x}$, 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. Finally, slightly abusing notation, let $X_i = \{j \in J \mid x_j = i\}$ denote the set of jobs that have chosen machine i in configuration $\boldsymbol{x}$, and define X_i^* analogously for $\boldsymbol{x^*}$.

A local policy for machine i uses only the information about the jobs on the same machine i, but it can look at all the parameters of these jobs, including their processing times on other machines. By contrast, a *strongly local* policy may depend only on the processing time that these jobs have on this machine i.

In order to quantify the inefficiency caused by the lack of coordination, we use the notion of *price of anarchy* [38], that is, the ratio between the social cost value of the worst Nash equilibrium and that of the social optimum. To be more precise, we are interested in upper bounds for the PoA of coordination mechanisms rather than the PoA of specific games. Applying [15] to the current context, the PoA of a coordination mechanism is defined to be the maximum ratio, taken over all the games G that the mechanism may induce, of the social cost of a Nash equilibrium of G divided by the optimum social cost achievable for the scheduling problem underlying G.

3   Deterministic Non-Preemptive Coordination Mechanisms

It is known that given an assignment of jobs to machines, in order to minimize the weighted sum of completion times, SmithRule is optimal [48]. It is therefore only natural to consider this policy as a good first candidate. 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 analysis uses the map $\varphi : I^J \to L_2([0,\infty))^I$, which maps a configuration to a vector of functions as follows. If $\boldsymbol{f} = \varphi(\boldsymbol{x})$, then

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

5

We let $\langle g, h \rangle := \int_0^\infty g(y)h(y)dy$ 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. We also define

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

We then have

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

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

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

The result follows. $\qquad\square$

Theorem 3.2. *The price of anarchy of SmithRule for unrelated machines $(R||\sum w_j c_j)$ is at most 4.*

Proof. Let $\boldsymbol{x}$ and $\boldsymbol{x}^*$ be two assignments, with $\boldsymbol{x}$ a Nash equilibrium, and write $\boldsymbol{f} = \varphi(\boldsymbol{x})$, $\boldsymbol{f}^* = \varphi(\boldsymbol{x}^*)$. We assume for simplicity that all jobs have distinct ratios (of processing time to weight). (This assumption is just for simplicity; alternatively, we could introduce a tie breaking rule.)

We first calculate a job j's completion time according to $\boldsymbol{x}$, and use the Nash condition:

$$c_j^{SR} = \sum_{\substack{k : x_k = x_j \\ \rho_{x_k k} < \rho_{x_j j}}} p_{x_k k} \; + \; p_{x_j j} \leq \sum_{\substack{k : x_k = x_j^* \\ \rho_{x_k k} < \rho_{x_j^* j}}} p_{x_k k} \; + \; p_{x_j^* j}.$$

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

6

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 $C^{SR}(\boldsymbol{x}) \leq 4C^{SR}(\boldsymbol{x}^*)$. $\qquad\square$

The following result, proved in Appendix A, shows that (assuming promptness) no deterministic non-preemptive strongly local mechanism can do better than SmithRule. This also implies that the bound of Theorem 3.2 is tight.

Theorem 3.3. *The pure PoA of any strongly local deterministic non-preemptive prompt coordination mechanism is at least 4. This is true even for the case of restricted identical machines $(B||\sum w_j c_j)$ with unweighted jobs.*

4 Improvements with Preemption and Randomization

4.1 Preemptive Coordination Mechanism

In this section, we study the power of preemption and present ProportionalSharing, a preemptive mechanism that is strictly better w.r.t. the PoA than any deterministic non-preemptive strongly local 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, using ProportionalSharing instead of SmithRule only increases the social cost[3] and doesn't decrease the cost of any player.

A better understanding of this result can be obtained by observing that in our context, preemptive policies can be thought of (and also implemented) as non-preemptive but also non-prompt policies. Jobs are run in an appropriate order, but possibly delayed past their completion time. (Notice however that such an implementation would technically disallow anonymous jobs, i.e., jobs that do not have IDs.) From this perspective, ProportionalSharing can be implemented by using SmithRule to determine the processing order, but then holding each job back after it is completed by an amount equal to the total delay it causes to other jobs SmithRule schedules after it. This fact can be seen explicitly in the first equation of the upcoming Lemma 4.1. In this way, the interests of a player are aligned with those of the group by having it "internalize its externalities", leading not only to better allocations but also to a better social cost, despite the extra charges. Additional advantages of this coordination mechanism are that, unlike SmithRule, it can handle anonymous jobs, and the games it induces always possess PNE.

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*

$$
\begin{aligned}
w_j c_j^{PS} &= \sum_{\substack{k \in X_i \setminus \{j\} \\ \rho_{ik} \leq \rho_{ij}}} w_j p_{ik} + \sum_{\substack{k \in X_i \\ \rho_{ik} > \rho_{ij}}} w_k p_{ij} + w_j p_{ij} \\
&= \sum_{k \in X_i \setminus \{j\}} w_j w_k \min\{\rho_{ik}, \rho_{ij}\} + w_j p_{ij}.
\end{aligned} \tag{1}
$$

---

[3]Note that this is not the case for the makespan social cost function.

Proof. We notice that the completion time of job j is only affected by the amount of "work" that the processor has completed by that time and not by the way this processing time has been shared among the jobs. For job j and any job k such that $\rho_{ik} \leq \rho_{ij}$, we know that their whole processing demands, p_{ij} and p_{ik} respectively, have been served. On the other hand, while job j is not complete, for each w_j units of processing time it receives, any job k with $\rho_{ik} > \rho_{ij}$ receives w_k units. Thus, when job j is completed, the processing time spent on any such job k will be exactly $\frac{p_{ij}w_k}{w_j}$. Adding all these processing times and multiplying by player j's weight, w_j gives the lemma. $\qquad\square$

Theorem 4.2. *The price of anarchy of* ProportionalSharing *for unrelated machines* $(R||\sum w_j c_j)$ *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. By Lemma 4.1, we see that for any assignment $\boldsymbol{x}$, $C^{PS}(\boldsymbol{x}) = \|\varphi(\boldsymbol{x})\|^2$; note the factor two difference compared to the first term for SmithRule. Moreover, (1) is upper bounded by

$$\sum_{k \in X_i} w_j w_k \min\{\rho_{ij}, \rho_{ik}\} + w_j p_{ij},$$

and so the Nash condition implies that for any equilibrium $\boldsymbol{x}$, and any other assignment $\boldsymbol{x}^*$,

$$C^{PS}(\boldsymbol{x}) \leq \sum_j \left(\sum_{k:x_k=x_j^*} w_j w_k \min\{\rho_{x_j^*j}, \rho_{x_j^*k}\} + p_{x_j^*j} \right)$$
$$= \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}^*) \rangle + \eta(\boldsymbol{x}^*).$$

This is identical to the equation we used (after an additional inequality) in the case of Smith's Rule.

Let $\boldsymbol{x}$ be a Nash assignment, $\boldsymbol{x}^*$ an optimal assignment w.r.t. Smith's Rule, and again define $\boldsymbol{f} = \varphi(\boldsymbol{x})$, $\boldsymbol{f}^* = \varphi(\boldsymbol{x}^*)$. Following the same method of analysis as for Smith's Rule, we obtain

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

using 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 [11], where in fact they show that even if C^{PS} is used for the cost of OPT, i.e., we consider the ratio $C^{PS}(\boldsymbol{x})/C^{PS}(\boldsymbol{x}^*)$, this can be arbitrarily close to $1 + \phi$. $\qquad\square$

In the case of equal weights, we obtain the following slightly improved bound. This result can be proven in our framework but using a variation of the Cauchy-Schwartz inequality derived from Lemma 4.4 below. However, we present here a different proof approach of independent interest.

Theorem 4.3. *The price of anarchy of* EqualSharing *for unrelated machines* $(R| |\sum c_j)$ *is at most* 2.5. *This bound is tight even for the restricted related machines model.*

Proof. We begin by proving the following lemma, which gives a tighter version of an inequality initially used by Christodoulou and Koutsoupias [14]:

8

Lemma 4.4. *For every pair of non-negative integers k and k^*,*

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

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

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

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

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

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

Now, using this lemma, we show that for any machine i:

$$\sum_{j \in X_i^*} c_j^{ES}(\boldsymbol{x}_{-j}, x_j^*) \leq \frac{1}{3}\sum_{j \in X_i} c_j^{ES}(\boldsymbol{x}) + \frac{5}{3}\sum_{j \in X_i^*} c_j^{SF}(\boldsymbol{x^*}).$$

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

For all the jobs $j \in Max_i$, we decrease p_{ij} by the minimum positive value Δ such that the cardinality of Max_i increases. After a change of this sort, the LHS drops by $(k^*(k+1))\Delta$ while the RHS drops by $(\frac{1}{3}k^2 + \frac{5}{3}\frac{k^*(k^*+1)}{2})\Delta$. Given Lemma 4.4 above, we conclude that the drop of the LHS is always less than or equal to the drop of the RHS. Using the same inequality again, we conclude that for unit jobs on machine i the inequality is always true; summing up over all $i \in I$ yields:

$$\sum_{j \in J} c_j^{ES}(\boldsymbol{x}_{-j}, x_j^*) \leq \frac{1}{3}\sum_{j \in J} c_j^{ES}(\boldsymbol{x}) + \frac{5}{3}\sum_{j \in J} c_j^{SF}(\boldsymbol{x^*}).$$

This gives a price of anarchy bound of 2.5.

The tightness of the bound follows from Theorem 3 of [11]. The authors present a load balancing game lower bound, which is equivalent to assuming that all jobs have unit size and the machines are using EqualSharing; thus the same proof yields a (pure) PoA lower bound for restricted related machines and unweighted jobs. □

On the negative side, we have the following (the proof of which can be found in Appendix A)

Proposition 4.5. *When jobs are anonymous, the worst-case PoA of any deterministic prompt coordination mechanism is at least $13/6$.*

9

4.2 Randomized Coordination Mechanism

In this section we examine the power of randomization and present Rand, which outperforms any prompt deterministic strongly local policy. Under Rand, for any pair of jobs on the same machine, the externalities they cause each other are shared equally in expectation. This is achieved with the following property: if two jobs j and j' are assigned to machine i, then

$$\mathbb{P}\{j \text{ precedes } j' \text{ in the ordering}\} = \frac{\rho_{ij'}}{\rho_{ij} + \rho_{ij'}}. \tag{2}$$

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 $\Omega(m)$ PoA bound for the makespan function [35], and the same family of examples developed in [35] gives an $\Omega(m)$ lower bound for this policy in our setting. Nevertheless, we will prove the following bounds:

Theorem 4.6. *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, this bound improves to $\pi/2$, which is tight.*

The high level approach for obtaining the upper bound is in exactly the same spirit as the previous section: find an appropriate mapping φ from an assignment into a convenient inner product space.

For simplicity, we assume in this section that the processing times have been scaled such that the ratios ρ_{ij} are all integral. This assumption is inessential and easily removed. We also take κ large enough so that, except for infinite processing times, $\rho_{ij} \leq \kappa$ for all $i \in I, j \in J$.

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 4.7. 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 $\varphi(\boldsymbol{x})_r^i$. Its value is then defined as

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

We also let $\varphi(\boldsymbol{x})^i$ denote the vector $(\varphi(\boldsymbol{x})_0^i, \varphi(\boldsymbol{x})_1^i, \ldots, \varphi(\boldsymbol{x})_\kappa^i)$.

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

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

Lemma 4.8. *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*

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

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

$$c_j^R = (M\boldsymbol{u}^i)_{\rho_{ij}} + p_{ij}.$$

10

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

$$c_j^R = \sum_{k \in X_i \setminus \{j\}} p_{ik} \mathbb{P}\{\text{job } k \text{ ahead of job } j\} + p_{ij}$$

$$= \sum_{k \in X_i \setminus \{j\}} p_{ik} \frac{\rho_{ij}}{\rho_{ij} + \rho_{ik}} + p_{ij}$$

$$= \sum_{k \in X_i} p_{ik} \frac{\rho_{ij}}{\rho_{ij} + \rho_{ik}} + \tfrac{1}{2} p_{ij}.$$

We can rewrite this in terms of the signature as

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

A crucial observation is the following:

Lemma 4.9. *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* [13], meaning that the determinant of any submatrix is positive. It follows immediately that H is positive definite, and hence so is M. $\qquad \square$

Thus we may define an inner product by

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle := \sum_{i \in I} (\boldsymbol{u}^i)^T M \boldsymbol{v}^i, \tag{3}$$

with an associated norm $\|\cdot\|$. 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

$$C^R(\boldsymbol{x}) = \|\varphi(\boldsymbol{x})\|^2 + \tfrac{1}{2} \eta(\boldsymbol{x}). \tag{4}$$

Competitiveness of Rand on a single machine. How well Rand performs on a *single* machine, compared to the optimal SmithRule, turns out to play an important role. So suppose we have n jobs with size p_j and weight w_j, for $j \leq n$. The signature $\boldsymbol{u}$ is given by just $u_r = \sum_{j : p_j / w_j = r} w_j$. Notice that the weighted sum of completion times according to SmithRule and Rand respectively are

$$\boldsymbol{u}^T S \boldsymbol{u} + \tfrac{1}{2} \sum_j w_j p_j \qquad \text{and} \qquad \boldsymbol{u}^T M \boldsymbol{u} + \tfrac{1}{2} \sum_j w_j p_j,$$

where $S_{rs} = \tfrac{1}{2} \min(r, s)$. 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 $\max_{\boldsymbol{u} \geq \boldsymbol{0}} \frac{\boldsymbol{u}^T M \boldsymbol{u}}{\boldsymbol{u}^T S \boldsymbol{u}}$. So the question is closely related to the worst-case distortion between two norms.

Interestingly, it turns out that this problem has been considered, and solved, in a different context. In [16], 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 4.10 ([16]). *For any sequence $u_1, u_2, \ldots, u_k$ with $u_r > 0$ for all r,*

$$\sum_{r,s} u_r u_s \frac{rs}{r+s} < \frac{\pi}{4} \sum_{r,s} u_r u_s \min\{r, s\}. \tag{5}$$

Moreover, this is tight [28] (take $p_j = 1/j^2$, $w_j = 1$, and let $n \to \infty$). We also present a quite different proof of the theorem in Appendix B. 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 4.11. *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. $\qquad\square$

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

Proof of Theorem 4.6. Let $\boldsymbol{x}$ be the assignment at a Nash equilibrium, and $\boldsymbol{x}^*$ the assignment of the optimal solution, and let $\boldsymbol{u} = \varphi(\boldsymbol{x})$ and $\boldsymbol{u}^* = \varphi(\boldsymbol{x}^*)$.

From the Nash condition and Lemma 4.8, we obtain

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

Applying Cauchy-Schwartz

$$\begin{aligned}
C^R(\boldsymbol{x}) &\leq \|\boldsymbol{u}^*\|\|\boldsymbol{u}\| + \eta(\boldsymbol{x}^*) \tag{6} \\
&\leq \tfrac{2}{3}\|\boldsymbol{u}^*\|^2 + \tfrac{3}{8}\|\boldsymbol{u}\|^2 + \eta(\boldsymbol{x}^*),
\end{aligned}$$

Now recalling the definition of φ and applying Lemma 4.11, 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 PoA of 32/15.

In the case where $\eta(\boldsymbol{x}^*)$ is negligible, we continue from (6):

$$
\begin{aligned}
C^R(\boldsymbol{x}) &\le \|\boldsymbol{u}^*\| \|\boldsymbol{u}\| \\
&\le \tfrac{1}{2}\|\boldsymbol{u}^*\|^2 + \tfrac{1}{2}\|\boldsymbol{u}\|^2 \\
&\le \tfrac{\pi}{4} C^{SR}(\boldsymbol{x}^*) + \tfrac{1}{2} C^R(\boldsymbol{x}),
\end{aligned}
$$

by Theorem 4.10 and Equation 4. Thus $C^R(\boldsymbol{x})/C^{SR}(\boldsymbol{x}^*) \le \pi/2$. $\qquad\square$

As noted in Appendix A, a slight modification of the construction used to prove Proposition 4.5 can be used to show that the worst-case PoA of Rand is at least $5/3$.

5 Existence of PNE and Algorithm

Existence of PNE. Under SmithRule it may happen that no pure Nash equilibrium exists [19]. Here we show that ProportionalSharing and Rand both induce exact potential games, which hence always have PNE. For the case of ProportionalSharing, this generalizes [22, Theorem 3], which addresses EqualSharing.

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

$$
\Phi^{PS}(\boldsymbol{x}) = \tfrac{1}{2} C^{PS}(\boldsymbol{x}) + \tfrac{1}{2} \eta(\boldsymbol{x}). \tag{7}
$$

Likewise, the Rand mechanism yields exact potential games with potential

$$
\Phi^R(\boldsymbol{x}) = \tfrac{1}{2} C^R(\boldsymbol{x}) + \tfrac{1}{2} \eta(\boldsymbol{x}). \tag{8}
$$

Proof. We give the proof for ProportionalSharing; the proofs for Rand and Approx are 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 $i' \ne i$.

We may write the change in the potential function as

$$
\Phi^{PS}(\boldsymbol{x}') - \Phi^{PS}(\boldsymbol{x}) = \sum_{k \in J} D_k \ + \ \tfrac{1}{2} w_j (p_{i'j} - p_{ij}), \tag{9}
$$

where

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

Consider a job $k \ne j$ 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

$$\sum_{k \in X_{i'}} D_k = \tfrac{1}{2} w_j \sum_{k \in X_{i'}} w_k \min\{\rho_{i'j}, \rho_{i'k}\}$$

$$= \tfrac{1}{2} w_j (c_j^{PS}(\boldsymbol{x'}) - p_{i'j}).$$

All other jobs are unaffected by the change, and so do not contribute to (9). 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. Finally we define Approx, a deterministic strongly local policy that we will use in order to design a combinatorial constant factor approximation algorithm for the underlying optimization problem. The completion time of a job in this policy is exactly its completion time if ProportionalSharing were being used plus its processing time, i.e. $c_j^A(\boldsymbol{x}) = c_j^{PS}(\boldsymbol{x}) + p_{x_j j}$.

Following the proof of Theorem 5.1 we can show that $\Phi^A(\boldsymbol{x}) = \tfrac{1}{2} C^A(\boldsymbol{x}) + \eta(\boldsymbol{x})$ is a potential function for the games induced by this mechanism, and following the proof of Theorem 4.2, the PoA of the mechanism is at most 4. The advantage of this mechanism is that $C^A(\boldsymbol{x}) = 2C^{SR}(\boldsymbol{x})$ for any configuration $\boldsymbol{x}$ and therefore, despite the larger PoA bound, computing an equilibrium allocation for the induced game yields a scheduling algorithm with approximation ratio 2 (because the scheduling algorithm, given the allocation, applies SmithRule and not Approx).

Computing such an allocation might be hard in general, but we show that imitating a natural best response dynamics gives rise to a simple polynomial time local search $(2 + \epsilon)$-approximation algorithm. At each iteration, the scheduling algorithm reassigns the job which thereby obtains the largest possible improvement in the Approx costing (a best response move). In other words, we use Approx in order to find a good allocation and then switch to SmithRule.

In order to bound the running time of our local-search algorithm we will use the β-nice concept of [3]. Given some configuration $\boldsymbol{x}$, let

$$\Delta(\boldsymbol{x}) = \sum_j \left(c_j(\boldsymbol{x}) - c_j(\boldsymbol{x}_{-j}, x_j') \right),$$

where x_j' is the best response to $\boldsymbol{x}_{-j}$ for player j. Awerbuch et al. [3] define an exact potential game with potential function Φ and social cost function C to be β-nice if and only if, for any configuration $\boldsymbol{x}$, both $\Phi(\boldsymbol{x}) \leq C(\boldsymbol{x})$ and $C(\boldsymbol{x}) \leq \beta OPT + 2\Delta(\boldsymbol{x})$ hold[4]. Among other dynamics, they consider what they call *basic dynamics*, where in each step, among all players that can uniquely deviate and improve their cost by some factor α, we choose the one with the largest absolute improvement, and allow that player to move. They subsequently show the following lemma, where $\boldsymbol{x}^*$ is the configuration that minimizes the potential function.

Lemma 5.2 ([3]). *Let $\tfrac{1}{8} > \epsilon > \alpha$. Consider an exact potential game that satisfies the β-nice property and any initial state $\boldsymbol{x}^0$. Then basic dynamics generates a profile $\boldsymbol{x}$ with $C(\boldsymbol{x}) \leq \beta(1 + O(\epsilon))OPT$ in at most $O\left(\frac{n}{\epsilon} \log\left(\frac{\Phi(\boldsymbol{x}^0)}{\Phi(\boldsymbol{x}^*)}\right)\right)$ steps.*

---

[4]In their definition, unlike ours, OPT denotes the optimum social cost w.r.t. the game's cost functions.

We define a *coordination mechanism* to be β-nice if all the games that it induces are β-nice with OPT being the optimum social cost of the underlying machine scheduling problem, independent of the coordination mechanism. Our next lemma shows that Approx satisfies these conditions.

Lemma 5.3. *The Approx coordination mechanism is β-nice with $\beta = 4$.*

Proof. It is easy to see that the potential function $\Phi^A(\boldsymbol{x}) = \frac{1}{2}C^A(\boldsymbol{x}) + \eta(\boldsymbol{x})$ satisfies $\Phi^A(\boldsymbol{x}) \le C^A(\boldsymbol{x})$ for all configurations $\boldsymbol{x}$, since $\eta(\boldsymbol{x}) \le \frac{1}{2}C^A(\boldsymbol{x})$. Therefore, what we need to show is that:

$$C^A(\boldsymbol{x}) \le \beta C^{SR}(\boldsymbol{x^*}) + 2\Delta(\boldsymbol{x}),$$

where

$$\Delta(\boldsymbol{x}) = \sum_{j \in J} \left(w_j c_j^A(\boldsymbol{x}) - w_j c_j^A(\boldsymbol{x}_{-j}, x_j') \right),$$

and x_j' is the best response for player j in configuration $\boldsymbol{x}$. We note that since $c_j^A(\boldsymbol{x}_{-j}, x_j') \le c_j^A(\boldsymbol{x}_{-j}, x_j^*)$, then:

$$C^A(\boldsymbol{x}) - \sum_{j \in J} w_j c_j^A(\boldsymbol{x}_{-j}, x_j^*) \le \Delta(\boldsymbol{x}).$$

Now following the same approach as in the proof of Theorem 4.2, we easily obtain that the PoA is at most 4. More specifically, we can obtain the inequality

$$\sum_{j \in J} w_j c_j^A(\boldsymbol{x}_{-j}, x_j^*) \le \tfrac{1}{4} C^A(\boldsymbol{x}) + 3C^{SR}(\boldsymbol{x^*}).$$

Summing these two inequalities and simplifying we obtain

$$C^A(\boldsymbol{x}) \le 4C^{SR}(\boldsymbol{x^*}) + \tfrac{4}{3}\Delta(\boldsymbol{x}),$$

proving the lemma. □

If we consider that every machine uses Approx, then, as a result of Lemma 5.3 along with Lemma 5.2 and the fact that $C^A(\boldsymbol{x}) = 2C^{SR}(\boldsymbol{x})$ for any configuration $\boldsymbol{x}$, we get the following theorem bounding the running time of our algorithm.

Theorem 5.4. *Starting from any initial configuration $\boldsymbol{x}^0$ and following basic dynamics leads to a profile $\boldsymbol{x}$ with $C^{SR}(\boldsymbol{x}) \le (2 + O(\epsilon))OPT$ in at most $O\left(\frac{n}{\epsilon} \log\left(\frac{\Phi^A(\boldsymbol{x}^0)}{\Phi^A(\boldsymbol{x^*})}\right)\right)$ steps.*

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 [31, 24, 36]. 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 approximation 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.

References

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

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

[3] 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.

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

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

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

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

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

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

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

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

[12] I. Caragiannis, C. Kaklamanis, and P. Kanellopoulos. Taxes for linear atomic congestion games. *ACM Transactions on Algorithms (to appear)*.

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

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

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

[16] 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.

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

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

[19] J.R. Correa and M. Queyranne. Efficiency of equilibria in restricted uniform machine scheduling with minsum social cost. 2010 (manuscript).

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

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

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

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

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

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

[26] 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.

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

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

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

[30] 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.

[31] M. Hoefer, V. S. Mirrokni, H. Röglin, and S.-H. Teng. Competitive routing over time. In *WINE*, pages 18–29, 2009.

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

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

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

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

[36] S. Koch and M. Skutella. Nash equilibria and the price of anarchy for flows over time. *Theory of Computing Systems*, to appear.

[37] 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.

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

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

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

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

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

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

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

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

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

[47] 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.

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

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

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

A Lower Bounds

Deterministic non-preemptive strongly local coordination mechanisms. Adapting a proof of Caragiannis et al. [11, Theorem 7], Correa and Queyranne [19, Theorem 13], showed that the pure PoA of games induced by SmithRule can be arbitrarily close to 4. This holds even with unweighted jobs in the restricted identical machines model (a similar construction in [24] for nonatomic prioritized selfish routing can also be easily adapted). Based on this construction, we show the same lowerbound for arbitrary strongly local prompt policies that are deterministic and non-preemptive. To aid in comprehension, we first demonstrate this for strongly local ordering policies (i.e., deterministic non-preemptive policies where the IIA property holds). We then discuss the changes needed to obtain the full result, but since the arguments are of a quite different flavour from the rest of the paper, we give only a sketch.

Proof of Theorem 3.3. We begin by presenting the family of game instances that leads to pure PoA approaching 4 for games induced by SmithRule in the restricted identical machines model [19], and then show how to generalize the lowerbound based on this construction.

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

If every job j_{xy} is assigned to machine y, there are exactly m/x^2 jobs with completion time x ($1 \le x \le k$), which leads to a total cost of $m \sum_{x=1}^{k} 1/x$. On the other hand, assigning each job to the machine with smallest index among all the ones that minimize its completion time gives a PNE assignment whose total cost is $\Omega(4m \sum_{x=1}^{k} 1/x)$ [19].

Ordering policies. We may of course modify the construction so that each job i can be assigned to only two machines: the machine O_j to which it is assigned under OPT, and the machine N_j it assigned to under the Nash (where $N_j \le O_j$ for all j). Since the job ordering under the optimal assignment does not affect the cost, we only need to make sure that for any jobs j, j' with $O_{j'} < O_j$, j gets higher priority than j' on O_j.

Given a specific lower bound instance for SmithRule, we have n job *slots*, each defined by the pair of machines O_j and N_j. 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 N_j machine index and we assign the first job of machine N_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 N_j, we first consider the slots with greater O_j machine index. This ensures that, given the PNE assignment, any job that deviates back to its OPT machine will suffer cost at least as much as in the SmithRule instance, while its cost in the PNE is the same as in the given instance. Therefore, the assignment of each job j to machine N_j is a PNE for this set of ordering policies.

Removing the IIA assumption. We modify the above construction to have N jobs, where N is extremely large, and one extra machine (so we have $M = m + 1$ machines). Each machine has an associated prompt policy (which may use job IDs); thus for any subset of jobs, the policy on a machine will specify the order that the jobs are run. We will then choose only a small subset of n jobs that will fill in the previously defined slots; the remaining jobs will all be assigned processing time 0 on machine $m + 1$, and infinity on all other machines; call such jobs *spurious*. By choosing the assignment of jobs to slots appropriately, we will be able to enforce the orderings we want on

the jobs, and obtain a Nash with the same cost as before. More precisely, we want the following, which ensures that the proposed Nash assignment is indeed an equilibrium:

(i) In the Nash assignment, the ordering on any machine is exactly as we require in the previously defined construction.

(ii) If we take the Nash assignment, but then any single job attempts to deviate, it will find itself at the back of the ordering. More carefully: if S_i is the set of jobs on machine i at Nash, and we consider any job j with $O_j = i$, then j is last according to the ordering determined by the set $S_i \cup \{j\}$ and the policy on machine i. This ensures that nobody has an incentive to deviate.

To prove this, we begin with the m-th machine, and argue that we can find a set $S_m \subset J$, with $|S_m|$ equal to the number of slots which have machine m as the Nash strategy, such that there is a very large set $Q_m \subset J$ with the following property:

Every job $j \in Q_m$ is last in the ordering on machine m determined by the set $S_m \cup \{j\}$.

We will assign S_m to the slots which run on machine m at OPT, and then make all jobs outside of S_m and J_m spurious. We then repeat this process on machine $m-1$, but selecting S_{m-1} and Q_{m-1} as subsets of Q_m. This construction guarantees an ordering satisfying properties (i) and (ii). The existence of the sets S_i and Q_i for all i follows from the following easily proved combinatorial lemma, asuming that N is chosen sufficiently large.

Lemma A.1. *Let k and r be integers, with $k > r$. For any subset S of $[k] := \{1, 2, ..., k\}$, let π_S be an ordering (permutation) of S, which may depend on S in an arbitrary manner, and define*

$$Q_S := \{j \in [k] \setminus S : j \text{ is last according to the order } \pi_{S \cup \{j\}}\}.$$

Then there exists a subset S of size r so that $|Q_S| \geq (k-r)/(r+1)$.

$\square$

Deterministic strongly local mechanisms. We give here a lower bound that applies to *any* deterministic prompt strongly local coordination mechanism, even when preemption is allowed, as long as jobs are anonymous.

Proof of Proposition 4.5. The construction is a slight variant of one given in Caragiannis et al. [11] 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 NE. 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 NE to the optimal cost converges to $13/6 > 2.166$. $\square$

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

B   The performance of Rand on a single machine

Proof of Theorem 4.10. 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 $x_1, x_2, \ldots, x_n$, $x_i \in \mathbb{N}$,
$$\sum_i \sum_j \frac{x_i x_j}{x_i + x_j} < \frac{\pi}{4} \sum_i \sum_j \min\{x_i, x_j\}. \tag{10}$$
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 (10) 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
$$\mu(i - 1/2) = 2 \sum_{j=1}^n \left(\frac{x_j}{x_i + x_j} \right)^2. \tag{11}$$
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 (11) 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 $\pi/2$. $\qquad\square$

C   A Reduction from Prioritized Selfish Routing

In this appendix, we show that in the unweighted case, and using ShortestFirst, the scheduling games under consideration form a special cases of the priority selfish routing games defined in [24]. This suffices to give upper bounds on the price of anarchy for ShortestFirst, and in fact the correct bound if the nonatomic case is considered.

A *priority selfish routing game* is defined as follows (except here, we will restrict ourselves to the unweighted case, where all players have unit demand). We are given a directed graph G, and a set of players $j = 1, \ldots, n$; each player has an associated source-sink pair (s_j, t_j), and must pick as their strategy some s_j-t_j path to route their demand.

Each arc e has an associated cost function f_e, which we will take to be linear; $f_e(x) = a_e x + b_e$. In the standard selfish routing game, the cost or delay experienced by a player using edge e is given by $f_e(x_e)$, where x_e is the load on the edge, i.e. (taking unit demands) the number of players using that edge. The total cost associated to an edge is then $x_e f_e(x_e)$. In the priority selfish routing model on the other hand, the total cost for an edge will be $\int_0^{x_e} f_e(z) dz$, the "area under the curve". This is split between the players using an edge, according to some ordering $\prec_e$ of the players using edge e: The t'th player in the ordering pays an amount $\int_{t-1}^t f_e(x) dx$. The ordering $\prec_e$ can be very general, and may depend on the strategies chosen by all the players (even those not using edge e).

In [24], it is shown that in this model, the price of anarchy is at most $17/3$ in the setting described above, which improves to 4 in the nonatomic case where any individual player is negligible. These upper bounds hold for any priority ordering.

We are now ready to describe the reduction. Begin with an instance of the scheduling game, with policy given by ShortestFirst. By scaling if necessary, assume that all finite p_{ij} satisfy $p_{ij} \leq 1$, and let Q be such that $Q \cdot p_{ij} \in \mathbb{N}$ for all finite p_{ij}. We construct a graph G as follows. There is a single sink node t which will be the destination for all players. Each machine i will correspond to a path P_i of length Q, and the cost function of each edge on the path will be simply $f_e(x) = x/Q$. Connect the end of each path to a common sink node t, with zero cost edges.

Now for each job j, we will have a source node s_j, and a player with source s_j and destination t. For each machine i, we add an arc from s_j to a node v_{ij} in the path corresponding to i, such that the fraction of the path between v_{ij} and the end of the path (towards t) is exactly p_{ij}. The cost of this arc will be a constant $p_{ij}/2$. To complete the definition of the priority selfish routing instance, we define the priority ordering on any edge in P_i according to ShortestFirst, in increasing order of p_{ij}.

There is a natural correspondence between an assignment in the scheduling problem and a routing in the priority routing problem. If job j uses machine i, then route j from s_j to v_{ij} and then to t.

Lemma C.1. *For any job j, the completion time c_j of the job in the scheduling instance is the same as the amount C_j player j pays in the derived priority routing instance.*

Proof. Suppose job j uses machine i. In the routing instance, all larger (w.r.t. processing time) jobs on the edge will not affect job j's cost, since shorter jobs have higher priority, All smaller jobs on the other hand will cause delays, on some subset of the edges on the path. In particular, a job k with $p_{ik} < p_{ij}$ causes a delay of 1 on a fraction p_{ik} of the edges on machine i's path. On an edge with ℓ players ahead of player j, j will pay an amount given by a trapezoidal area: $\frac{1}{2Q}(\ell + (\ell + 1)) = \frac{\ell + 1/2}{Q}$. Summing up the costs over all edges used by j, we get

$$C_j = p_{ij}/2 + \sum_{k : p_{ik} < p_{ij}} p_{ik} + p_{ij}/2 = p_{ij} + \sum_{k : p_{ik} < p_{ij}} p_{ik}. \qquad \square$$

It follows immediately that the Nash equilibria of the scheduling game and the derived priority routing coincide, and that social costs are also the same. Thus the worst-case price of anarchy of the unweighted scheduling game is no worse than the worst-case price of anarchy in the unweighted priority routing model, i.e., 17/3.