Decentralized Throughput Scheduling*

Jasper de Jong, Marc Uetz, and Andreas Wombacher

University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
{j.dejong-3, m.uetz, a.wombacher}@utwente.nl

Abstract. Motivated by the organization of distributed service systems, we study models for throughput scheduling in a decentralized setting. In throughput scheduling, a set of jobs j with values w_j, processing times p_{ij} on machine i, release dates r_j and deadlines d_j, is to be processed non-preemptively on a set of unrelated machines. The goal is to maximize the total value of jobs scheduled within their time window $[r_j, d_j]$. While approximation algorithms with different performance guarantees exist for this and related models, we are interested in the situation where subsets of machines are governed by selfish players. We give a universal result that bounds the price of decentralization: Any local α-approximation algorithm, $\alpha \geq 1$, yields Nash equilibria that are at most a factor $(\alpha+1)$ away from the global optimum, and this bound is tight. For identical machines, we improve this bound to $\sqrt[\alpha]{e}/(\sqrt[\alpha]{e}-1) \approx (\alpha + 1/2)$, which is shown to be tight, too. The latter result is obtained by considering subgame perfect equilibria of a corresponding sequential game. We also address some variations of the problem.

1 Model and Notation

We consider a non-preemptive scheduling problem with unrelated machines, to which we refer as *decentralized throughput scheduling problem* throughout the paper. The input of an instance $I \in \mathcal{I}$ consists of a set of jobs $\mathcal{J}$, a set of machines $\mathcal{M}$, and a set of players $\mathcal{N}$. Each job $j \in \mathcal{J}$ comes with a release time r_j, a deadline d_j, a nonnegative value w_j and a processing time p_{ij} if scheduled on machine $i \in \mathcal{M}$. Machines can process only one job at a time. Job j is feasibly scheduled (on any of the machines) if its processing starts no earlier than r_j and finishes no later than d_j. For any set of jobs $S \subseteq \mathcal{J}$, we let $w(S) = \sum_{j \in S} w_j$ be the total value. Each player $n \in \mathcal{N}$ controls a subset of machines $M_n \subseteq \mathcal{M}$ and aims to maximize the total value of jobs that can be feasibly scheduled on its set of machines M_n. Here M_n, $n \in \mathcal{N}$, is a partition of the set of machines $\mathcal{M}$.

In this paper we are interested in *equilibrium allocations*, which we define as an allocation in which none of the players n can improve the total value of jobs that can be feasibly scheduled on its set of machines M_n by removing some of its jobs and adding some of the yet unscheduled jobs. Here we make the assumption that a player cannot make a claim on jobs that are scheduled on machines of

* Research supported by CTIT, Centre for Telematics and Information Technology, University of Twente, project "Mechanisms for Decentralized Service Systems".

other players. An equilibrium allocation is a (pure) Nash equilibrium (*NE*) in a strategic form game where player n's strategies are all subsets of jobs $S_n \subseteq \mathcal{J}$. If jobs S_n can be feasibly scheduled on machines M_n, then player n's valuation for S_n is $w(S_n) = \sum_{j \in S_n} w_j$, and we let $w(S_n) = -\infty$ otherwise. Furthermore, the utility of player n is $-\infty$ whenever S_n is not disjoint with the sets chosen by all other players. This way, in any strategy profile $(S_n)_{n \in \mathcal{N}}$ that is at Nash equilibrium, the sets S_n, $n \in \mathcal{N}$, are pairwise disjoint.

Our main focus will be the analysis of the price of decentralization, better known as the price of anarchy (PoA) [11], lower bounding the quality of any Nash equilibrium relative to the quality of a globally optimal allocation, *OPT*. Here *OPT* is an allocation maximizing the weighted sum of feasibly scheduled jobs over all players. More specifically, we are interested in the ratio

$$\text{PoA} = \sup_{I \in \mathcal{I}} \sup_{NE \in NE(I)} \frac{w(OPT)}{w(NE)}, \tag{1}$$

where $NE(I)$ denotes the set of all Nash equilibria of instance I. Note that *OPT* is a Nash equilibrium too, hence the price of stability, as proposed in [1], equals 1.

In general, the question whether a strategy profile $(S_n)_{n \in \mathcal{N}}$ is a Nash equilibrium describes an NP-hard optimization problem for each player, even if each player controls a single machine only [14]. Therefore, we also consider a relaxed equilibrium condition: We say an allocation is an α-approximate Nash equilibrium (α-*NE*) if none of the players n can improve the total value of jobs that can be feasibly scheduled on its set of machines M_n by a factor larger than α by removing some of its jobs and adding some of the yet unscheduled jobs. By the existence of constant factor approximation algorithms for (centralized) throughput scheduling, e.g. [3, 4], the players are thus equipped with polynomial time algorithms to reach an α-*NE* in polynomial time, for certain constant values α.

As an interesting variant of the model described thus far, we also propose to analyze the price of anarchy for subgame perfect equilibria of an extensive form game as introduced by Selten [12, 17]. Here, we make the assumption that players select their subsets of jobs sequentially in an arbitrary but fixed order. In that situation, the n-th player is presented the set of yet unscheduled jobs $\mathcal{J} - \bigcup_{i<n} S_i$, from which he may select a subset S_n once, and is not allowed to revoke this decision later. For the special case where all machines are identical, the resulting subgame perfect equilibria of the extensive form game are provably better than Nash equilibria of the strategic game.

2 Motivation, Related Work and Contribution

Our motivation to study this problem is to analyze the performance of decentralized service systems, where jobs are posted, e.g. on a portal, and service providers can select these on a take-it-or-leave-it basis. The problem can be seen as a stylized version of coordination problems that appear in several application domains. We give three examples: (1) When operating microgrids for decentralized energy production and consumption, the goal is to consume locally produced energy as much as possible. Here, jobs can be defined as the operation of appliances (e.g. operating a washing machine), bounded by a time window and attached with a certain \$-value. Machines, on the other side, are local energy producers like PV-panels or micro CHPs [2, 15]. (2) In cloud computing, service providers such as Amazon and Google provide an infrastructure service, that is, provide a virtual machine with a specific service level for a certain period of time. The aim of a federated cloud computing environment, e.g. [6], is to "coordinate load distribution among different cloud-based data centers in order to determine optimal location for hosting application services ". (3) In private car sharing portals like Tamyca or Autonetzer [19], clients post car rental requests for a certain time period, and the price they are willing to pay. Car owners in the vicinity can select requests and rent their car(s). Stripping off the online nature from these applications exactly yields the type of problems we address.

The underlying non-strategic optimization problem is sometimes referred to as throughput scheduling. See for example [3], and follow-up papers, e.g. [4]. In the 3-field notation of [9], the problem reads $\mathrm{R}|r_j|\sum w_j U_j$, where R denotes the unrelated machine model, r_j specifies that there are release dates, and the objective is to minimize the total weight of late jobs. In terms of the optimal objective value this is equivalent to the maximization objective considered here, yet it is standard to revert to the maximization version for the purpose of approximation. Indeed, approximation algorithms for several versions of the maximization problem have been discussed in the literature, e.g., with or without weights, identical or unrelated machines, most notably [3, 4]. Special cases that are of particular interest are the single machine case with unit weights and zero release dates, solved in polynomial time by the Moore-Hodgson algorithm [16], and the case with identical machines and unit processing times, which can be cast and solved as an assignment problem [5]. To the best of our knowledge, the decentralized version that we propose here has not been addressed before.

Our contribution lies in the informal claim that the price of decentralization is very moderate: If local decisions of all players are approximately optimal with performance guarantee α, then any equilibrium allocation is not worse than an $(\alpha+1)$-fraction of the global optimum. We improve this to $\approx(\alpha+1/2)$ when all machines are identical, and when we consider only subgame perfect equilibria of a corresponding extensive form game. Along the way, we also obtain some additional insights.

3 A First Encounter

Example 1. There are two players $\mathcal{N}=\{1,2\}$, each controlling exactly one of two related machines $\mathcal{M}=\{1,2\}$, with machine speeds $s_1=1, s_2=\frac{2}{3}$, respectively[1]. There are two jobs $\mathcal{J}=\{1,2\}$ with processing times $p_1=p_2=1$, deadlines $d_1=1, d_2=\frac{3}{2}$ and values $w_1=w_2=1$. Release dates are $r_1=r_2=0$. ◁

In this example, when job 1 is allocated to machine 1 and job 2 to machine 2, both jobs can meet their respective deadlines. This is obviously an optimal

[1] This is a special case of the unrelated machine model by letting $p_{ij}=p_j/s_i$.

allocation. However when job 2 is allocated to machine 1, only one job can be scheduled before its deadline. See also Figure 1. Note that both allocations are

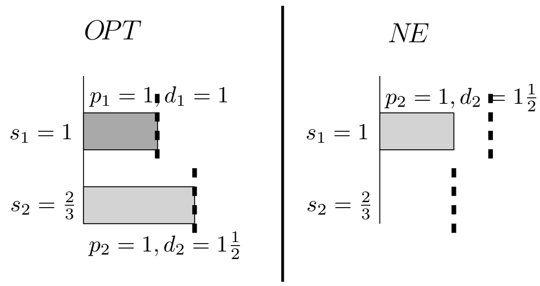

Fig. 1. Optimal solution and Nash equilibrium in the case of related machines.

a Nash equilibrium. Now $w(OPT)/w(NE) = 2/1 = 2$ for the second allocation, and we see from this simple example that

$$\text{PoA} \geq 2$$

in (1), even for the case of related machines, unit weights, unit processing times and zero release dates. The strategic form game for Example 1 with both Nash equilibria in boldface is shown in Figure 2. A corresponding extensive form game

		player 2			
		$\emptyset$	{1}	{2}	{1,2}
player 1	$\emptyset$	0,0	$0, -\infty$	$0, 1$	$0, -\infty$
	{1}	1,0	$-\infty, -\infty$	$\mathbf{1, 1}$	$-\infty, -\infty$
	{2}	$\mathbf{1, 0}$	$1, -\infty$	$-\infty, -\infty$	$-\infty, -\infty$
	{1,2}	$-\infty, 0$	$-\infty, -\infty$	$-\infty, -\infty$	$-\infty, -\infty$

Fig. 2. Strategic form game for Example 1 with Nash equilibria.

where players select their jobs sequentially, player 1 first, and suppressing the solutions for the trivially inferior strategies $\{1, 2\}$, is shown in Figure 3. Note that each subgame perfect equilibrium of this extensive form game yields an allocation that corresponds to a Nash equilibrium of the strategic form game. Yet the extensive form game has generally more Nash equilibria (here, 3) due to richer strategy spaces of players.

4 Bounds for Approximate Equilibrium Allocations

The players problem to decide if a strategy is at equilibrium is polynomially solvable only for special cases. For instance when jobs have unit values and zero

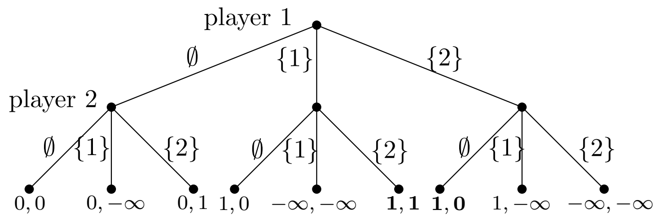

Fig. 3. Extensive form game for Example 1 with subgame perfect equilibria.

release dates, and when each player controls exactly one machine, the Moore-Hodgson algorithm [16] maximizes the total number of early jobs. But when players control more than one machine, the players problem is NP-complete as generalization of the makespan minimization problem on parallel machines [8]. When the machines M_n of a player n are identical, and jobs have unit processing times, the players' problem can be cast and solved as an assignment problem [5]. In most other cases, the players' problem is NP-complete. For example, for a player that controls a single machine, when jobs have zero release dates, but arbitrary processing times and weights, the problem is (weakly) NP-hard [13, 10]. Adding nontrivial release dates makes the problem strongly NP-hard [14].

Therefore, we consider a relaxed equilibrium concept, assuming that players strategies are only approximately optimal. This leads to the concept of α-approximate equilibria, which has lately been discussed also in the literature on computing Nash equilibria, for instance in the context of congestion games [18]. Approximate Nash equilibria can also be defined by allowing additive deviations instead of relative deviations, e.g. [7], but given that there exist constant-factor approximation algorithms for throughput scheduling, e.g. [3, 4], it appears more reasonable to work with relative bounds here. We say the allocation is an α-approximate Nash equilibrium, or α-*NE*, if no player n can improve the total value of its jobs by a factor larger than α. That said, we obtain the following.

Theorem 1. *The decentralized throughput scheduling problem has $PoA = \alpha + 1$, assuming that equilibrium allocations are α-approximate Nash equilibria. The lower bound $PoA \geq \alpha + 1$ even holds for the special case of unit values w_j, unit processing times p_j, related machines and zero release dates.*

Proof. First we prove PoA $\leq \alpha + 1$. Take any instance with optimal solution OPT and Nash equilibrium NE[2], and let NE_n and OPT_n, $n \in \mathcal{N}$, be the jobs allocated to player n in NE and OPT, respectively. For any $S \subseteq \mathcal{J}$, let $\overline{S} = \mathcal{J} \setminus S$ be the complement of S in $\mathcal{J}$.

Since all jobs in $\overline{NE}$ are available, and all jobs in OPT_n can be feasibly be scheduled by player n, by the definition of α-approximate Nash equilibrium, we have for all n, $\alpha w(NE_n) \geq w(OPT_n \cap \overline{NE})$. Now we get, by using linearity of

[2] In a slight abuse of notation, we use OPT and NE to also denote the set of feasibly scheduled jobs in the respective solutions.

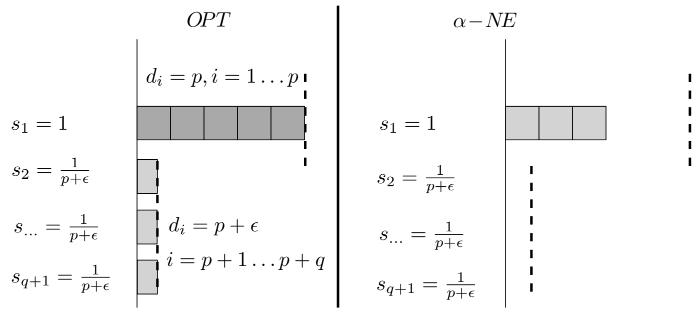

Fig. 4. Optimal solution and α-*NE* in case of related machines.

the objective function across players,

$$\begin{aligned}(\alpha+1)w(NE) &\geq \alpha w(NE) + w(OPT \cap NE)\\ &= \sum\nolimits_n \alpha w(NE_n) + w(OPT \cap NE)\\ &\geq \sum\nolimits_n w(OPT_n \cap \overline{NE}) + w(OPT \cap NE)\\ &= w(OPT)\,.\end{aligned}$$

To prove PoA $\geq \alpha + 1$ we give a tight example.

Example 2. Consider an instance with unit processing times $p_j = 1$, unit values $w_j = 1$, related machines, and zero release dates. Assume w.l.o.g. that $\alpha = p/q$, $p \geq q$, and assume players deploy an α-approximation each. There are $q+1$ players $\mathcal{N}$, each controlling one of $q+1$ machines $\mathcal{M} = \{1, \dots, q+1\}$ with machine speeds $s_1 = 1$ and $s_2 = s_3 = \dots = s_{q+1} = 1/(p+\varepsilon)$ for some $0 < \varepsilon < 1$. There are $p+q$ jobs $\mathcal{J} = \{1, \dots, p+q\}$. Jobs $J_1 = \{1, \dots, p\}$ have deadline p. Jobs $J_2 = \{p+1, \dots, p+q\}$ have deadline $p+\varepsilon$. ◁

Here, machine 1 can schedule at most p jobs. Machines $2, \dots, q+1$ can schedule no jobs from J_1 and only one job from J_2 each. In *OPT* all $p+q$ jobs are feasibly scheduled: jobs J_1 on Machine 1 and each of machines $2, \dots, q+1$ has one job from J_2. Now consider the α-approximate Nash equilibrium where only q jobs are scheduled: Machine 1 schedules all q jobs from J_2, and machines $2, \dots, q+1$ schedule no job. This is indeed an α-approximate Nash equilibrium, as machine 1 can schedule at most $p = \alpha q$ jobs, and since all jobs from J_2 are scheduled on machine 1, machines $2, \dots, q+1$ cannot improve from their 0 jobs either. See Figure 4 for an illustration. We conclude that PoA $\geq (p+q)/q = \alpha + 1$. □

Note that $\alpha = 1$ in the special case where the players can verify if a solution is a Nash equilibrium; in that case PoA $= 2$. Also note that the given upper bound is universal in the sense that it is independent of how the (α-approximate) Nash equilibrium is obtained. It is conceivable that specific algorithms can yield a better bound for the price of anarchy. However, the existence of more complicated counter-examples for specific algorithms is not unlikely either, and we did not take the effort to find them.

5 Subgame Perfect Equilibria

We here propose[3] to analyze the extensive form game in which the players select their subsets of jobs sequentially, and are not allowed to revoke their decisions later. Following Selten [17], an equilibrium of an extensive form game is called (α-approximate) subgame perfect if it induces a (α-approximate) Nash equilibrium in every subgame. The following example shows that indeed, not all (α-approximate) Nash equilibria are (α-approximate) subgame perfect.

Example 3. There are n players each controlling one of n identical machines $\mathcal{M} = \{1, \ldots, n\}$, and $2n-1$ jobs $\mathcal{J} = \{1, \ldots, 2n-1\}$ with unit weights. Jobs $J_1 = \{1, \ldots, n\}$ have processing time $1/n$ and deadline 1. Jobs $J_2 = \{n+1, \ldots, 2n-1\}$ have processing time 1 and deadline 1. $\triangleleft$

In *OPT*, machine 1 schedules jobs J_1 and machines $2, \ldots, n$ schedule jobs J_2. Consider Nash equilibrium *NE* where each machine schedules one job from J_1. Note that *NE* is indeed an equilibrium: no machine can schedule more than one job without exchanging jobs with another machine. See Figure 5 for an illustration. For this instance $w(OPT)/w(NE) = \frac{2n-1}{n} \to 2$ for $n \to \infty$. This Nash

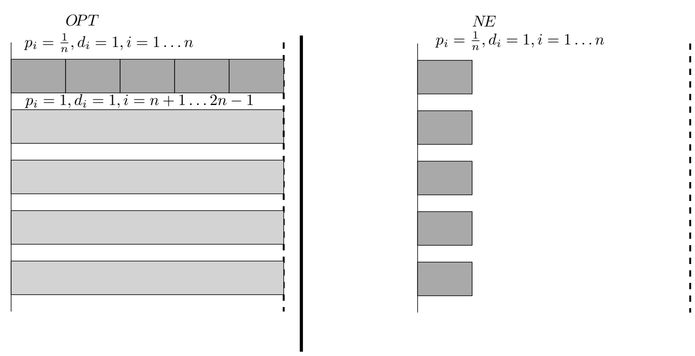

Fig. 5. An optimal solution and a Nash equilibrium in case of identical machines.

equilibrium is not subgame perfect, however. In any subgame perfect equilibrium, the first player would necessarily schedule all jobs from J_1 on his machine.

This example also shows that the identical machine model does not allow an improvement of the result of Theorem 1. Although non-subgame perfect equilibria might seem unrealistic, the equilibrium obtained in this example is quite

[3] Note added in proof: The idea to analyze subgame perfect equilibria of extensive form games rather than Nash equilibria of strategic form games has been proposed also by Paes Leme, R., Syrgkanis, V. and Tardos, É. in: The Curse of Simultaneity, Proceedings ITCS, pp. 60-67, ACM (2012).

reasonable: In a round robin assignment, each player chooses to schedule the most flexible available job(s) first.

Remark: It is not hard to see that each subgame perfect equilibrium of the sequential game proposed here corresponds to an outcome equivalent Nash equilibrium of the (non-sequential) strategic form game that we studied in Section 4. In that sense, our move to subgame perfect equilibria indeed makes sense from the perspective of worst case analysis.

6 Identical Machines

In this section we improve our previous results for the special case of identical machines when considering (α-approximate) subgame perfect equilibria of an extensive form game in which players select their jobs sequentially in any order.

6.1 Identical Machines: Lower Bound

We give a lower bound on the price of anarchy for subgame perfect equilibria.

Theorem 2. *$PoA \geq \sqrt[\alpha]{e}/(\sqrt[\alpha]{e}-1)$ for identical machines, even in the restricted model where we only consider α-approximate subgame perfect equilibria, and for unit processing times, unit weights, and zero release dates.*

Proof. We give a corresponding example.

Example 4. There are n players controlling one of n identical machines $\mathcal{M} = \{1, \ldots, n\}$. There are n^2 jobs $\mathcal{J} = \{1, \ldots, n^2\}$ with unit processing times and unit weights. Jobs have deadlines $\delta \in \{1, \ldots, n\}$ and for each deadline, there are n jobs with this deadline, that is, for all δ, $d_j = \delta$ for $j = 1 + (\delta - 1)n, \ldots, \delta n$.$\triangleleft$

We refer to jobs as δ-jobs, $\delta = 1, \ldots, n$. In Figure 6 we see an instance and solution for $n = 5$ and $\alpha = 2$ (that is, machines use a 2-approximation). For each of the jobs, the number displayed on it corresponds to its deadline. In OPT, every machine schedules n jobs with different deadlines, ordered by increasing deadline. Therefore $w(OPT) = n^2$. We construct an α-approximate subgame perfect equilibrium, say S, as follows. For every machine $i = 1, \ldots, n$ in this order, we find the maximum number of jobs that can be scheduled, say o_i, and let S_i be the $\lceil o_i/\alpha \rceil$ jobs with the largest deadlines (which are the most flexible jobs). For example, for $n = 5$ and $\alpha = 2$, $w(S) = 3 + 3 + 2 + 2 + 2 = 12$ as can be seen in Figure 6. We bound $w(S)$ in the following way. In S, denote by $r_\delta(i)$ the fraction of δ-jobs on machine i, relative to the total number of jobs on machine i. Let $r_\delta = \sum_i r_\delta(i)$. In our example, $r_4 = 0 + \frac{1}{3} + 1 + 1 + 0$. Observe that $\sum_\delta r_\delta = n$ for any allocation. In S, any machine scheduling a δ-job, does not schedule any job with deadline $(\delta + 2)$ or larger, hence it schedules at most $\lceil (\delta + 1)/\alpha \rceil \leq (\delta + 1 + \alpha)/\alpha$ jobs. Therefore, each job with deadline δ contributes at least $\alpha/(\delta + 1 + \alpha)$ to r_δ. For any δ for which all n δ-jobs are allocated in S, we get $r_\delta \geq n\alpha/(\delta + 1 + \alpha)$.

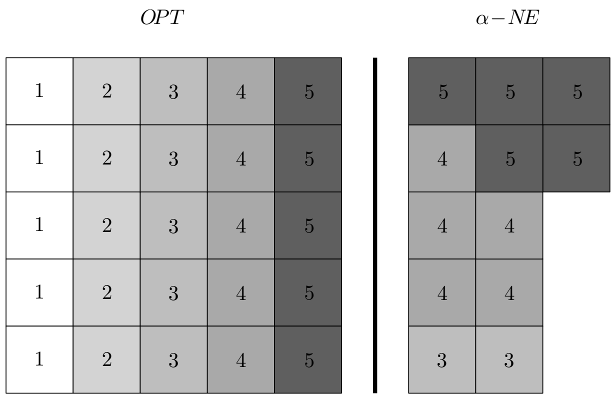

Fig. 6. Optimal solution and 2-approximate subgame perfect equilibrium in case of identical machines. Numbers denote job deadlines.

Now, for some $\delta' \geq 0$, by construction of the allocation we have that all n δ-jobs with $\delta = n - \delta', \ldots, n$ are fully scheduled, as well as a fraction of the $(n - (\delta' + 1))$-jobs. We get

$$n \geq \sum_{\delta=n-\delta'}^{n} r_\delta \geq \sum_{\delta=n-\delta'}^{n} \frac{n\alpha}{\delta + 1 + \alpha} \geq \int_{\delta=n-\delta'}^{n} \frac{n\alpha}{\delta + 1 + \alpha} d\delta \,. \tag{2}$$

Because the last term is upper bounded by n, we can derive an upper bound on δ'. In fact, basic calculus shows that

$$\delta' > \frac{(n+1+\alpha)(\sqrt[\alpha]{e} - 1)}{\sqrt[\alpha]{e}} \quad \Rightarrow \quad \int_{\delta=n-\delta'}^{n} \frac{n\alpha}{\delta + 1 + \alpha} d\delta > n \,,$$

which together with (2) yields that $\delta' \leq \frac{(n+1+\alpha)(\sqrt[\alpha]{e}-1)}{\sqrt[\alpha]{e}}$. Because only δ-jobs with $\delta \geq n - (\delta' + 1)$ are scheduled, we conclude that

$$w(S) \leq (\delta' + 1)n \leq \frac{(n + 1 + \alpha + \frac{\sqrt[\alpha]{e}}{\sqrt[\alpha]{e}-1})(\sqrt[\alpha]{e} - 1)}{\sqrt[\alpha]{e}} \cdot n \,.$$

We see that

$$\frac{w(OPT)}{w(S)} \geq \frac{n\sqrt[\alpha]{e}}{(n + 1 + \alpha + \frac{\sqrt[\alpha]{e}}{\sqrt[\alpha]{e}-1})(\sqrt[\alpha]{e} - 1)} \to \frac{\sqrt[\alpha]{e}}{\sqrt[\alpha]{e} - 1} \quad \text{for } n \to \infty \,,$$

and the claim follows. □

Note that the lower bound construction assumes that players choose the most flexible jobs first, which seems reasonable. The bound also holds for the case with unit processing times, where we may assume that the players use optimal strategies [5], that is $\alpha = 1$. For that case, the result shows that the price of anarchy can be as high as $e/(e-1) \approx 1.58$.

6.2 Identical Machines: Upper Bound

To derive a matching upper bound for identical machines, when considering only subgame perfect equilibria, we use a proof idea from Bar-Noy et al. [3] in their analysis of k-GREEDY, but need a nontrivial generalization to make it work for the case where players control multiple machines.

Assume there are n players and m identical machines, and each player i controls m_i machines. Denote by S_i the set of jobs selected by player i, and $S = \bigcup_i S_i$ the total set of jobs scheduled in an α-approximate subgame perfect equilibrium. The following lemma lower bounds the total weight collected by i.

Lemma 1. *We have for all players* i

$$w(S_i) \geq \frac{m_i}{m\alpha} w\left(OPT\left(\mathcal{J} \setminus \bigcup\nolimits_{j<i} S_j\right)\right) .$$

where $OPT(W)$ *denotes an optimal solution for any given set of jobs* W *and* m *machines.*

Proof. Let $W := \mathcal{J} \setminus \bigcup_{j<i} S_j$. Let OPT_i denote the maximum weight set of jobs that can be scheduled by player i. Observe that $w(OPT_i) \geq (m_i/m)OPT(W)$. This follows because player i could potentially select the jobs scheduled on the m_i most valuable machines from $OPT(W)$, as all machines are identical. Now, by definition $w(S_i) \geq w(OPT_i)/\alpha \geq m_i w(OPT(W))/(m\alpha)$. Here, the first inequality holds because we assume an α-approximate Nash equilibrium, and in particular no player will choose a subset of jobs that is not disjoint from the subsets selected earlier. □

We are now ready to prove the following.

Theorem 3. *PoA* $\leq \sqrt[\alpha]{e}/(\sqrt[\alpha]{e}-1)$ *for identical machines and* α*-approximate subgame perfect equilibria.*

Proof. Due to space limitations, we skip some technicalities of the proof, but give the main idea here. Let $\gamma := m\alpha$, and recall that $w(OPT) = w(OPT(\mathcal{J}))$ denotes the value of the optimal solution. We use Lemma 1, to get

$$w(S_i) \geq \frac{m_i}{\gamma} w\left(OPT\left(\mathcal{J} \setminus \bigcup\nolimits_{j<i} S_j\right)\right) \geq \frac{m_i}{\gamma}\left(w(OPT) - \sum\nolimits_{j<i} w(S_j)\right) ,$$

where the latter inequality holds because $w(OPT) - \sum_{j<i} w(S_j)$ represents the value of a feasible solution for the jobs $\mathcal{J} \setminus \bigcup_{j<i} S_j$. Add $\sum_{j=1}^{i-1} w(S_j)$ to both sides to get

$$\sum_{j=1}^{i} w(S_j) \geq \frac{m_i w(OPT)}{\gamma} + \frac{\gamma - m_i}{\gamma} \sum_{j=1}^{i-1} w(S_j) . \tag{3}$$

We prove by induction on i that

$$\sum_{j=1}^{i} w(S_j) \geq \frac{\gamma^{m_i'} - (\gamma-1)^{m_i'}}{\gamma^{m_i'}} w(OPT) ,$$

where $m'_i = \sum_{j=1}^{i} m_j$. When $i = 1$, we can show by induction on m_1 that $w(S_1) \geq \frac{\gamma^{m_1}-(\gamma-1)^{m_1}}{\gamma^{m_1}} w(OPT)$. Assume the claim holds for $i-1$. Applying the induction hypothesis to (3) we get

$$\sum_{j=1}^{i} w(S_j) \geq \frac{m_i w(OPT)}{\gamma} + \frac{\gamma - m_i}{\gamma} \cdot \frac{\gamma^{m'_{i-1}} - (\gamma-1)^{m'_{i-1}}}{\gamma^{m'_{i-1}}} w(OPT)\,.$$

This can be used to prove the inductive claim, using basic but careful calculus. Hence we get for $i = n$ (see also [3, Thm 3.3])

$$w(S) = \sum_{j=1}^{n} w(S_j) \geq \frac{\gamma^m - (\gamma-1)^m}{\gamma^m} w(OPT)\,.$$

We get

$$\mathrm{PoA} \leq \frac{\gamma^m}{\gamma^m - (\gamma-1)^m} = \frac{(m\alpha)^m}{(m\alpha)^m - (m\alpha-1)^m} \leq \frac{\sqrt[\alpha]{e}}{\sqrt[\alpha]{e}-1}\,, \tag{4}$$

where the the last inequality follows because the right hand side is exactly the limit for $m \to \infty$, and the series $b_m = (m\alpha)^m/((m\alpha)^m - (m\alpha-1)^m)$ is monotone in m, with $b_1 = \alpha \leq \sqrt[\alpha]{e}/(\sqrt[\alpha]{e}-1)$. □

Theorems 2 and 3 yield $\mathrm{PoA} = \sqrt[\alpha]{e}/(\sqrt[\alpha]{e}-1)$ when considering only α-approximate subgame perfect equilibria. Basic calculus shows that

$$\alpha + \frac{1}{2} \leq \sqrt[\alpha]{e}/(\sqrt[\alpha]{e}-1) \leq \alpha + \frac{1}{e-1}$$

for $\alpha \geq 1$. Also, for $\alpha \to \infty$ this value approaches $\alpha + \frac{1}{2}$. Note that for $\alpha = 1, \mathrm{PoA} = e/(e-1) \approx 1.58$.

Concluding Remarks

We briefly mention some more results for the the case $\alpha = 1$, that is, the case of Nash equilibrium allocations. Due to space limitations, any details are deferred to a full version of this paper.

First, we can show that the bound $\mathrm{PoA} = \sqrt{e}/(\sqrt{e}-1)$ for identical machines with unit processing times, unit weights and zero release dates holds without requiring that the Nash equilibria are subgame perfect. Next, we can generalize our results to a setting where bundle costs are not additive: When $w(J) \neq \sum_{j\in J} w_j$, but if we know that that $\sum_{j\in J} w_j/\beta \leq w(J) \leq \beta \sum_{j\in J} w_j$ for all $J \subseteq \mathcal{J}$ and for some parameter $\beta \geq 1$, then we can show that $\mathrm{PoA} = \beta^4 + \beta^2$. (Note that $\beta^4 + \beta^2 = 2$ for $\beta = 1$.) Also, when we allow players to afterwards trade one single job, or even a set of jobs (for money), we can show that this does not improve the PoA substantially.

The most challenging next step from an application viewpoint is to consider online settings. When the goal is (constant) competitive ratios for online-time models, however, we will most probably need to revert to preemptive scheduling models.

Acknowledgements. Thanks to Johann Hurink for some helpful discussions on the context, and Rudolf Müller and Frits Spieksma for very helpful remarks.

References

1. E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. In: Proc. 45th Symp. Foundations of Computer Science (FOCS), 295–304, IEEE, 2004.
2. V. Bakker, M.G.C. Bosman, A. Molderink, J.L. Hurink and G.J.M. Smit. Demand side load management using a three step optimization methodology. In: 1st IEEE Int. Conf. on Smart Grid Communications, 431-436. IEEE, 2010.
3. A. Bar-Noy, S. Guha, J. Naor, and B. Schieber. Approximating the throughput of multiple machines in real-time scheduling. *SIAM J. Comp.*, 31(2): 331–52, 2001.
4. P. Berman and B. Dasgupta. Multi-phase Algorithms for Throughput Maximization for Real-Time Scheduling. J. Combinatorial Optimization, 4(3): 307-323, 2000.
5. P. Brucker. Scheduling Algorithms (4th ed.), Springer Verlag, Berlin, 2004
6. R. Buyya, R. Ranjan, and R. Calheiros. Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. In C.-H. Hsu, L. Yang, J. Park, and S.-S. Yeo (eds.) *Algorithms and Architectures for Parallel Processing*, Lecture Notes in Computer Science 6081, 13-31. Springer, 2010.
7. C. Daskalakis, A. Mehta and C. Papadimitriou. Progress in Approximate Nash Equilibria. In: Proceedings 8th ACM EC, 355-358. ACM, 2007.
8. M.R. Garey and D.S. Johnson. "Strong" NP-completeness results: motivation, examples, and implications. Journal of the ACM, 25(3):499-508, 1978.
9. R. Graham, E. Lawler, J. Lenstra, and A. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5(2):287–326, 1979.
10. R.M. Karp. Reducibility among combinatorial problems. In: Complexity of computer computations, R.E. Miller and J.W. Thatcher (eds.), 85-103, 1972.
11. E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In: Proceedings 16th STACS, Lecture Notes in Computer Science 1563, 404–413. Springer, 1999.
12. H. W. Kuhn. Extensive Games and the Problem of Information, Contribution to the Theory of Games, Vol. II, Annals of Math. Studies, 28, 193-216, 1953
13. E.L. Lawler and J.M. Moore. A functional equation and its application to resource allocation and sequencing problems. Management Science, 16:77-84, 1969.
14. J.K. Lenstra, A.H.G. Rinnooy Kan, and P. Brucker. Complexity of Machine Scheduling Problems. Annals Disc. Math., 1:343-362, 1977.
15. A. Molderink, V. Bakker, M.G.C. Bosman, J.L. Hurink and G.J.M. Smit. Management and Control of Domestic Smart Grid Technology. IEEE transactions on Smart Grid, 1(2): 109-119, 2010.
16. J.M. Moore. An n job, one machine sequencing algorithm for minimizing the number of late jobs. Management Science, 15:102-109, 1968.
17. R. Selten. A simple model of imperfect competition, where 4 are few and 6 are many, International Journal of Game Theory, Vol. 2, 141-201, Physica Verlag 1973
18. E. Skopalik and B. Vöcking. Inapproximability of pure Nash equilibria. In: Proc. 40th Symp. on Theory of Computing (STOC), 355-364, ACM, 2008.
19. `www.tamyca.com` and `www.autonetzer.com`