The Price of Envy-Freeness in Machine Scheduling*

Vittorio Bilò[1], Angelo Fanelli[2], Michele Flammini[3], Gianpiero Monaco[3], and Luca Moscardelli[4]

[1] University of Salento, Italy
vittorio.bilo@unisalento.it
[2] CNRS, (UMR-6211), France
angelo.fanelli@gmail.com
[3] Gran Sasso Science Institute, L'Aquila, Italy
{michele.flammini,gianpiero.monaco}@univaq.it
[4] University of Chieti-Pescara, Italy
luca.moscardelli@unich.it

Abstract. We consider k-envy-free assignments for scheduling problems in which the completion time of each machine is not k times larger than the one she could achieve by getting the jobs of another machine, for a given factor $k \geq 1$. We introduce and investigate the notion of price of k-envy-freeness, defined as the ratio between the makespan of the best k-envy-free assignment and that of an optimal allocation achievable without envy-freeness constraints. We provide exact or asymptotically tight bounds on the price of k-envy-freeness for all the basic scheduling models, that is unrelated, related and identical machines. Moreover, we show how to efficiently compute such allocations with a worsening multiplicative factor being at most the best approximation ratio for the minimum makespan problem guaranteed by a polynomial time algorithm for each specific model. Finally, we extend our results to the case of restricted assignments and to the objective of minimizing the sum of the completion times of all the machines.

1 Introduction

The evolution of scheduling closely tracked the development of computers. Given m machines that have to process n jobs, minimizing the makespan of an assignment of the jobs to the machines is one of the most well-studied problem in the Theory of Algorithms [12,16,17,19]. In more details, assuming that the processing of job i on machine j requires time $p_{ij} > 0$, the completion time of machine j (under a certain assignment) is given by the sum of the processing times of all the jobs allocated to j. The makespan of an assignment is the maximum completion time among all the machines (we stress that an assignment is not forced to use all the available machines) and the objective of the scheduling problem is to find an assignment of minimum makespan.

* This research was partially supported by the PRIN 2010–2011 research project ARS TechnoMedia (Algorithmics for Social Technological Networks), funded by the Italian Ministry of University and Research.

In the literature, three different models of machines have been adopted. The general setting illustrated above is called scheduling problem with *unrelated machines* [19]. An interesting particular scenario is the case with *related machines* [17], where each job i has a load $l_i > 0$ and each machine j has a speed of processing $s_j > 0$, and thus the processing time of job i on machine j is given by $p_{ij} = l_i/s_j$. Finally, the even more specific setting in which the speed of each machine is 1 is referred to as the scheduling problem with *identical machines* [12,16]. Even this latter problem is NP-hard [16].

The approximability of the scheduling problem has been well understood for all the three models described above. However, all the proposed solutions do not envisage fair allocations in which no machine prefers (or envies) the set of the tasks assigned to another machine, i.e., for which her completion time would be strictly smaller. In the literature, such fairness property is referred to as "envy-freeness" [8,9]. Specifically, consider a scenario in which a set of tasks (jobs) has to be allocated among employees (machines) in such a way that the last task finishes as soon as possible. It is natural to consider fair allocations, that is allocations where no employee prefers (or envies) the set of tasks assigned to some other employee, i.e., a set of tasks for which her completion time would be strictly smaller than her actual one.

It is possible to consider two different variants of this model, depending on the fact that an employee (i) can envy the set of tasks assigned to any other employee or (ii) can only envy the set of tasks of other employees getting at least one job: in the latter case, employees not getting any job do not create envy. In the following, we provide some scenarios motivating both variants.

For the first variant, consider a company that receives an order of tasks that must be assigned among its m employees. For equity reasons, in order to make the workers satisfied with their task assignment so that they are as productive as they can, the tasks should be assigned in such a way that no envy is induced among the employees.

For the second variant, consider a scenario in which a company, in order to fulfill a complex job composed by several tasks, has to engage a set of employees that, for law or trade union reasons have to be all paid out the same wage. Again, for making the workers as productive as they can, it is required that no envy is induced, but in this case we are interested only in the envy among the *engaged* employees, i.e. the ones receiving at least a task to perform.

We notice that the existence of envy-free schedules is not guaranteed in the first variant of the model. For instance, consider a scenario where the number of machines is strictly greater than the number of jobs. Clearly at least one machine would not get any job and all the machines getting at least one job would be envious. Therefore, in the following of this paper we focus on the second varant of the model, in which envy-freeness is required only among machines getting at least one job.

We adopt a more general definition of envy-free allocations, namely the *k-envy-freeness* (for any $k \geq 1$): Given an assignment and two machines j, j' (where both j and j' get jobs), we say that j k-envies j' if the completion time of j

is at least k times the completion time she would have when getting the set of jobs assigned to j'. In other words, an assignment is k-envy-free if no machine would decrease her completion time by a factor at least k by being assigned all the jobs allocated to another machine. Notice that a k-envy-free assignment always exists: a trivial one can be obtained by allocating all the jobs to a single machine, even if it might have a dramatically high makespan.

We are interested in analyzing the loss of performance due to the adoption of envy-free allocations. Our study has an optimistic nature and, then, aims at quantifying the efficiency loss in the best k-envy-free assignment. Therefore, we introduce the **price of k-envy-freeness**, defined as the ratio between the makespan of the best k-envy-free assignment and that of an optimal assignment. In the literature, other papers performed similar optimistic studies, see, for instance, [1,6]. The price of k-envy-freeness represents an ideal limitation to the efficiency achievable by any k-envy-free assignment. In our work, we also show how to efficiently compute k-envy-free assignments which nicely compare with the performance of the best possible ones. We point out that the computation of non-trivial k-envy-free assignments is necessary to achieve good quality solutions, since the ratio between the makespan of the worst k-envy-free assignment and that of an optimal assignment can be very high. In particular, it is unbounded for unrelated machines, $n\frac{s_{max}}{s_{min}}$ for related ones, where s_{max} (resp. s_{min}) is the maximum (resp. minimum) speed among all the machines, and n for identical machines.

Related Work. The scheduling problem with unrelated machines has been studied in [19]. The authors provide a 2-approximation polynomial time algorithm and show that the problem cannot be approximated in polynomial time within a factor less than 3/2. Polynomial time approximation schemes for related and identical machines have been presented in [17] and [16], respectively.

The problem of fair allocation is a longstanding issue, thus, the literature on this topic includes hundreds of references. For a nice review, we refer the reader to the book [4]. One common notion of fairness, recurring in many papers and therefore adopted for central problems, is that of envy-freeness. For instance, the classical Vickrey auctions [23], as well as some optimal Bayesian auctions [2,20], generate envy-free outcomes. An interesting paper explicitly dealing with envy-free auctions is [13]. Studies on envy-free divisions, typically referred to as envy-free cake cutting, can be found in [3,8,9]. Furthermore, [10,14] consider algorithmic issues related to the envy-free pricing problem, that is a scenario in which a seller has to set (envy-free) prices and allocations of items to buyers in order to maximize the total revenue.

Concerning scheduling problems, an important stream of research is the one focusing on envy-free algorithmic mechanism design. Roughly speaking, algorithmic mechanism design is the attempt of motivating the machines, through payments or incentives, to follow desired behaviors (*truthful mechanisms*). Upper and lower bounds on the approximation ratio achieved by truthful mechanisms have been given in [7,18,22]. However, such papers are not concerned with fair allocations. To the best of our knowledge, envy-free mechanisms for the scheduling problem with unrelated machines have been first considered in [15]. the authors prove a lower bound of $2 - 1/m$ and an upper bound of $(m + 1)/2$ on the performance guarantee of envy-free truthful mechanisms. Such upper and lower bounds have been improved in [5] to $O(\log m)$ and $\Omega\left(\frac{\log m}{\log\log m}\right)$, respectively. Recently, [11] shows that no truthful mechanism can guarantee an envy-free allocation with a makespan less than a factor of $O(\log m)$ the optimal one, thus closing the gap. It is worth noticing that, for $k = 1$, our model can be seen as a special case of the one considered in [5,11,15] when the same payment is provided to all the machines receiving at least a job, while no payment is given to the other machines.

The work most closely related to our study is [6]. The authors consider the envy-free scheduling problem with unrelated machines with some substantial differences with respect to our setting. Specifically, *i)* they only consider 1-envy-free assignments (while we consider k-envy-free assignments, for any $k \geq 1$); *ii)* the objective in their work is that of minimizing the sum of the completion times of all jobs (while we mainly consider the makespan); *iii)* in their setting all the machines contribute to create envy (while in our setting only machines getting at least one job are considered for the envy-freeness). Not surprisingly, the authors prove that, in their setting, the price of envy-freeness is unbounded.

Our Results. We consider the price of k-envy-freeness in the scheduling problem, that is, the ratio between the makespan of the best k-envy-free assignment and that of an optimal assignment. We investigate the cases of unrelated, related and identical machines and provide exact or asymptotically tight bounds on the price of k-envy-freeness. We stress that low values of k implies a greater attitude to envy, which tremendously reduces the set of k-envy-free assignments. A natural threshold that arose in our analysis of the cases with related and identical machines is the value $k = 2$, as it can be appreciated in the following table where we summarize our main results. They are fully described in Section 3.

		Identical	Related	Unrelated
$k = 1$	UB and LB	$\min\{n, m\}$	$\min\{n, m\}$	$2^{\min\{n,m\}-1}$
$k \in (1, 2)$	UB	$\frac{2k}{k-1}$	$2k\sqrt{\frac{m}{k-1}}$	$\left(1 + \frac{1}{k}\right)^{\min\{n,m\}-1}$
	LB	$\Omega\left(\frac{2k}{k-1}\right)$	$\Omega\left(\sqrt{\frac{m}{k-1}}\right)$	$\left(1 + \frac{1}{k}\right)^{\min\{n,m\}-1}$
$k \geq 2$	UB	$1 + \frac{1}{k}$	$2 + \max\left\{1, \sqrt{\frac{m}{k}}\right\}$	$\left(1 + \frac{1}{k}\right)^{\min\{n,m\}-1}$
	LB	$1 + \frac{1}{k}$	$\max\left\{1, \sqrt{\frac{m}{k}}\right\}$	$\left(1 + \frac{1}{k}\right)^{\min\{n,m\}-1}$

A further result derives from the fact that our upper bound proofs are constructive and, therefore, they de facto provide polynomial time algorithms able to calculate good k-envy-free assignments. Such an extension is discussed in Section 3.4. Furthermore, in Subsection 4.1 we also consider the *restricted* scheduling problem, where each job can be assigned only to a subset of machines. Moreover, besides considering the problem of minimizing the makespan, we consider in Subsection 4.2 the problem of minimizing the sum of the completion times of all the machines.

Due to space constraints, some proofs are omitted.

2 Preliminaries

In the scheduling problem, there are $m \geq 2$ *machines* and n indivisible *jobs* to be assigned to the machines. In the *unrelated* case, the time of running job i on machine j is given by $p_{ij} > 0$. In the *related* setting, each job i has a load $l_i > 0$, each machine has a speed of processing $s_j > 0$, and the processing time of job i on machine j is given by $p_{ij} = l_i/s_j$. We refer to the specific setting in which the speed of each machine is 1 as *identical*, where $p_{ij} = l_i$.

For an integer $h > 0$, define $[h] := \{1, \ldots, h\}$. In the related and identical setting, we denote with $L = \sum_{i \in [n]} l_i$ the total load of all the jobs and with $l_{max} = \max_{i \in [n]} l_i$ the maximum load of a job.

An *assignment* or *solution* $\mathbf{N}$ is specified by a partition of the set of jobs into m components, i.e., $(N_j)_{j \in [m]}$, where N_j denotes the set of jobs assigned to machine j. Let Q be a set of jobs, we use the notation $C_j(Q)$ to denote the completion time of machine j on the set Q, i.e., $C_j(Q) = \sum_{i \in Q} p_{ij}$. Thus $C_j(N_j)$ denotes the completion time of machine j under the assignment $\mathbf{N}$. For the related and identical settings, let $L_j(\mathbf{N})$ be the total load of the jobs assigned by $\mathbf{N}$ to machine $j \in [m]$, i.e., $L_j(\mathbf{N}) = \sum_{i \in N_j} l_i$ and $L_{min}(\mathbf{N}) = \min\{L_j(\mathbf{N}) : j \in [m] \wedge N_j \neq \emptyset\}$ (resp. $L_{max}(\mathbf{N}) = \max\{L_j(\mathbf{N}) : j \in [m] \wedge N_j \neq \emptyset\}$) the minimum (resp. maximum) load of the non-empty machines in $\mathbf{N}$. Notice that, in the related setting, we have $C_j(N_j) = L_j(\mathbf{N})/s_j$ and, for the identical one, $C_j(N_j) = L_j(\mathbf{N})$. The *makespan* of assignment $\mathbf{N}$ is defined as $\mathcal{M}(\mathbf{N}) = \max_{j \in [m]} C_j(N_j)$, that is the maximum processing time among all the machines. An *optimal* assignment is one minimizing the makespan. We denote by $\mathbf{O}$ an optimal assignment.

Given an assignment $\mathbf{N}$, a real value $k \geq 1$, and two machines j, j' such that $N_j \neq \emptyset$ and $N_{j'} \neq \emptyset$, we say that j k-**envies** j' if $C_j(N_j) > kC_j(N_{j'})$. An assignment $\mathbf{N}$ is k-**envy-free** if $C_j(N_j) \leq kC_j(N_{j'})$ for every pair of machines (j, j') such that $N_j \neq \emptyset$ and $N_{j'} \neq \emptyset$. Notice that a k-envy-free assignment can always be obtained by assigning all jobs to a single machine. The **price of** k-**envy-freeness** (PoEF$_k$) is defined as the ratio between the makespan of the best k-envy-free assignment and the makespan of an optimal assignment. More formally, let $\mathcal{F}_k$ be the set of the k-envy-free assignments, then PoEF$_k$ = $\min_{\mathbf{N} \in \mathcal{F}_k} \frac{\mathcal{M}(\mathbf{N})}{\mathcal{M}(\mathbf{O})}$.

We conclude this section with some preliminary general results.

Proposition 1. *For the scheduling problem with related machines,* PoEF$_k \leq \min\{n, m\}$ *for any* $k \geq 1$.

Proof. Assume that machine 1 is the fastest one, i.e., $s_1 \geq s_j$ for each $j \in [m]$. Clearly, the solution $\mathbf{N}$ assigning all jobs to machine 1 is k-envy-free for any $k \geq 1$ and has $\mathcal{M}(\mathbf{N}) = \frac{L}{s_1} \leq \frac{nl_{max}}{s_1}$. By $\mathcal{M}(\mathbf{O}) \geq \frac{l_{max}}{s_1}$ and $\mathcal{M}(\mathbf{O}) \geq \frac{L}{ms_1}$, we obtain the claim. □

Such a simple upper bound on the price of k-envy-freeness proves to be tight when $k = 1$ even for the setting of identical machines.

Proposition 2. *For the scheduling problem with identical machines, there exists an instance for which* PoEF$_k = \min\{n, m\}$ *when* $k = 1$.

We now show that, for finite values of k, a price of k-envy-freeness equal to 1 cannot be achieved even in the setting of identical machines.

Proposition 3. *For the scheduling problem with identical machines, no value of k (possibly depending on n and m) can guarantee* $\mathrm{PoEF}_k = 1$.

In the next lemma, we give an important result which helps to characterize the performance of k-envy-free solutions in the case of related machines.

Lemma 1. *For a value $k \geq 1$, an instance of the scheduling problem with related machines, and an integer $2 \leq h \leq \min\left\{m, \left\lfloor \frac{L(k-1)}{kl_{max}} \right\rfloor\right\}$, there always exists a k-envy-free solution $\mathbf{N}$ using exactly h machines and such that $\mathcal{M}(\mathbf{N}) \leq \frac{L/h+l_{max}}{s_h}$, where s_h is the speed of the h-th fastest machine.*

3 Results

3.1 Identical Machines

In this subsection, we consider the scheduling problem with identical machines. For the case of $k \geq 2$, we can prove a constant upper bound on the price of k-envy freeness.

Theorem 1. *For the scheduling problem with identical machines,* $\mathrm{PoEF}_k \leq 1 + 1/k$ *for any $k \geq 2$.*

Proof. We argue that applying Algorithm 1 to any initial assignment $\mathbf{S}$, we get a k-envy free assignment $\mathbf{N}$ with makespan at most $\mathcal{M}(\mathbf{S})(1 + 1/k)$. The claim follows by choosing as the starting assignment $\mathbf{S}$ an optimal solution $\mathbf{O}$.

Initially Algorithm 1 manipulates the starting assignment in such a way that it becomes an assignment with makespan 1 with the minimal number of non-empty machines, and such that the machines are numbered so that to a smaller index corresponds a larger or equal load. After the first phase we assume that the jobs are assigned to machines in $[\overline{m}]$.

Since machine $\overline{m}$ is the least loaded one, if $L_{\overline{m}}(\mathbf{S}) \geq 1/k$, then $\mathbf{S}$ is k-envy-free and the claim follows. On the other side, if $L_{\overline{m}}(\mathbf{S}) < 1/k$, we move all the jobs in $\mathbf{S}$ from machine $\overline{m}$ to machine $\overline{m}-1$ obtaining a new assignment $\mathbf{N}$ which is k-envy-free. In fact, in the new assignment $\mathbf{N}$, machine $\overline{m}-1$ gets a load larger than 1, thus becoming the most loaded machine, whereas any other machine has a load smaller than 1. Machine $\overline{m}-1$ does not envy any other machine, since $L_{\overline{m}-1}(\mathbf{N}) = L_{\overline{m}-1}(\mathbf{S}) + L_{\overline{m}}(\mathbf{S}) \leq 2L_{\overline{m}-1}(\mathbf{S}) \leq kL_{\overline{m}-1}(\mathbf{S}) \leq kL_j(\mathbf{N})$, for each $j \leq \overline{m}-1$ and $k \geq 2$. Thus, we can conclude that the new assignment is k-envy-free. Finally we see that the makespan of $\mathbf{N}$ is at most $L_{\overline{m}-1}(\mathbf{N}) = L_{\overline{m}-1}(\mathbf{S}) + L_{\overline{m}}(\mathbf{S}) \leq L_{\overline{m}-1}(\mathbf{S}) + 1/k \leq (1 + 1/k)L_{\overline{m}-1}(\mathbf{N}) \leq \mathcal{M}(\mathbf{S})(1 + 1/k)$. The claim follows. $\square$

The next result shows that the above upper bound is tight for any $k \geq 2$.

Algorithm 1.

```
1: Input: assignment $\mathbf{S}$
2: Rescale the loads in such a way that $\mathcal{M}(\mathbf{S}) = 1$
3: while there exists a pair of machines $(j, j')$ s.t. $L_j(\mathbf{S}) + L_{j'}(\mathbf{S}) \leq 1$ do
4:     $S_j \leftarrow S_j \cup S_{j'}$
5:     $S_{j'} \leftarrow \emptyset$
6: end while
7: Renumber the machines in non-increasing order of loads
         $L_j(\mathbf{S}) \geq L_{j+1}(\mathbf{S})$ for each $j \in [m-1]$
8: Let $[\overline{m}]$ be the set of machines with at least one job assigned
9: Create a new assignment $\mathbf{N}$ defined as follows
10: if $L_{\overline{m}}(\mathbf{S}) < 1/k$ then
11:     $N_j \leftarrow S_j$ for each $j < \overline{m} - 1$
12:     $N_{\overline{m}-1} \leftarrow S_{\overline{m}-1} \cup S_{\overline{m}}$
13:     $N_j \leftarrow \emptyset$ for each $j > \overline{m} - 1$
14: else
15:     $N_j \leftarrow S_j$ for each $j \in [m]$
16: end if
17: return $\mathbf{N}$
```

Proposition 4. *For the scheduling problem with identical machines, given any $k \geq 2$, there exists an instance for which* $\text{PoEF}_k \geq 1 + 1/k - \epsilon$, *for any* $\epsilon > 0$.

For the remaining case of $k \in (1, 2)$, the following bounds hold.

Theorem 2. *For the scheduling problem with identical machines,* $\text{PoEF}_k \leq \min\left\{\frac{2k}{k-1}, n, m\right\}$ *for any* $k \in (1, 2)$.

Theorem 3. *For the scheduling problem with identical machines, given any* $k \in (1, 2)$, *there exists an instance for which* $\text{PoEF}_k = \Omega\left(\min\left\{\frac{2k}{k-1}, n, m\right\}\right)$.

3.2 Related Machines

In this subsection, we consider the scheduling problem with related machines.

Theorem 4. *For the scheduling problem with related machines,* $\text{PoEF}_k \leq 2 + \max\left\{1, \sqrt{\frac{m}{k}}\right\}$ *for any* $k \geq 2$.

Proof. Given an instance of the scheduling problem with related machines, consider any assignment $\mathbf{S}$. Let us normalize the machine speeds and the loads of the jobs so that the fastest machine has speed 1 and the makespan of solution $\mathbf{S}$ is 1, i.e., $\mathcal{M}(\mathbf{S}) = 1$. Let us rename the machines in such a way that $s_j \geq s_{j+1}$ for any $j = 1, \ldots, m-1$; notice that $L_1(\mathbf{S}) \leq 1$ and we can assume that $L_j(\mathbf{S}) \geq L_{j+1}(\mathbf{S})$ for any $j = 1, \ldots, m-1$ (otherwise by swapping S_j and S_{j+1} a solution having equal or better makespan could be obtained).

Denote by $M_1 = \{1, \ldots, |M_1|\}$ the set of machines having load at least $1/k$ in $\mathbf{S}$, i.e., $L_j(\mathbf{S}) \geq 1/k$ for any $j \in M_1$, and by M_2 the set of the remaining

machines. Note that it also holds $s_j \geq 1/k$ for any $j \in M_1$. Moreover, it is easy to check that no pair (j, j') of machines in M_1 is such that j k-envies j'.
In the following we build a new allocation $\mathbf{N}$ starting from allocation $\mathbf{S}$.

Let L_j^i be the load of each machine j at the moment in which the job i is considered for allocation by Algorithm 2. The new assignment $\mathbf{N}$ is obtained as described in Algorithm 2.

Algorithm 2.

```
1: Input: assignment $\mathbf{S}$
2: $\mathbf{N} \leftarrow \mathbf{S}$
3: $M' \leftarrow \emptyset$
4: $j' \leftarrow |M_1|$
5: while $j' < m$ do
6:     $j \leftarrow j' + 1$
7:     if $k > m$ or $\sum_{p \geq j} L_p(\mathbf{S}) \leq \sqrt{\frac{m}{k}}$ then
8:         for each job in $i \in \bigcup_{p \geq j} S_p$ do
9:             Let $j'' \in M_1 \cup M'$ be the machine with the current smallest load
10:            if $L_1^i + l_i \leq k L_{j''}^i$ then
11:                $N_1 \leftarrow N_1 \cup \{i\}$                ▷ Assign job $i$ to machine 1
12:            else
13:                $N_{j''} \leftarrow N_{j''} \cup \{i\}$          ▷ Assign job $i$ to machine $j''$
14:            end if
15:        end for
16:        $j' \leftarrow m$
17:    else
18:        $M' \leftarrow M' \cup \{j\}$
19:        Let $j'$ such that $\frac{1}{k} - L_j(\mathbf{S}) \leq \sum_{p=j+1}^{j'} L_p(\mathbf{S}) \leq \frac{2}{k} - L_j(\mathbf{S})$
20:        $N_j \leftarrow \bigcup_{p=j}^{j'} S_p$
21:    end if
22: end while
23: return $\mathbf{N}$
```

Assignment $\mathbf{N}$ is initially set equal to assignment $\mathbf{S}$. When lines 18–20 are executed, it means that $k \leq m$ and $\sum_{p \geq j} L_p(\mathbf{S}) > \sqrt{\frac{m}{k}}$. It follows that $L_j(\mathbf{S}) > \frac{1}{\sqrt{mk}}$ (and therefore also $s_j > \frac{1}{\sqrt{mk}}$). In this case, the only machine receiving some new jobs is machine j. Since the load of any machine in M_2 is less than $1/k$, we can gather all the jobs of machines $j+1, j+2, \ldots, j'$ of total load between $\frac{1}{k} - L_j(\mathbf{S})$ and $\frac{2}{k} - L_j(\mathbf{S})$, and add in $\mathbf{N}$ all such jobs to machine j. We obtain $C_j(N_j) = \frac{L_j(\mathbf{N})}{s_j} \leq \frac{\frac{2}{k}}{\frac{1}{\sqrt{m}\sqrt{k}}} = 2\sqrt{\frac{m}{k}}$. Notice that machine j cannot be k-envied by any other machine, and (since $k \geq 2$) cannot k-envy other machines with load at least $1/k$ (all the machines in $M_1 \cup M'$ have load at least $1/k$ in assignment $\mathbf{N}$).

Note that lines 8–16 can be executed only once. When they are executed, it means that $k > m$ or $\sum_{p \geq j} L_p(\mathbf{S}) \leq \sqrt{\frac{m}{k}}$. When $k > m$, the load in $\mathbf{S}$ of each machine in M_2 is at most $1/m$ and the total load of all machines in M_2 is at most 1. Therefore, in any case, the total load of all machines $p \geq j$ is at most $\max\left\{1, \frac{\sqrt{m}}{\sqrt{k}}\right\}$. Notice that, since *(i)* $k \geq 2$, *(ii)* the load of each machine in M_1 is at least $1/k$ already in allocation $\mathbf{S}$ and *(iii)* the load of each job to be assigned is at most $1/k$, it is always possible to maintain k-envy-free an allocation by assigning each job either to machine 1 or (in case the assignment to machine 1 would result in a state non being k-envy-free) to the machine of $M_1 \cup M'$ having the smallest load at that moment. In fact, consider any job i belonging in $\mathbf{S}$ to some machine $p \geq j$, and, for any $j \in M_1 \cup M'$, let L_j^i be the load of machine j at the moment in which the job i is considered for assignation by Algorithm 2. Assigning job i to machine j'' results in a k-envy-free state because $L_{j''}^i + l_i \leq kL_{j''}^i$ as conditions *(i)*, *(ii)* and *(iii)* hold.

Let us now compute the makespan of assignment $\mathbf{N}$, by considering only the machines receiving some new jobs in lines 8–16 of Algorithm 2:

The total load added to machine 1 is at most $\max\left\{1, \frac{\sqrt{m}}{\sqrt{k}}\right\}$ and therefore the total load $C_1(N_1)$ of machine 1 at the end of the process is at most $1 + \max\left\{1, \frac{\sqrt{m}}{\sqrt{k}}\right\}$.

For any machine $j \in M_1 \setminus \{1\}$, let $last(j)$ be the last job assigned to machine j and $\ell_j = l_{last(j)}$ its load. Since $last(j)$ has not been assigned to machine 1, it must hold that $L_1^{last(j)} + \ell_j > kL_{j'}^{last(j)}$ for some $j' \in M_1 \setminus \{1\}$. In particular, $L_1^{last(j)} + \ell_j > kL_j^{last(j)}$ because $last(j)$ has been assigned to the machine with minimum load at that moment. Since the total load that can be given to machine 1 is at most $1 + \max\left\{1, \sqrt{\frac{m}{k}}\right\}$, it follows that $L_j^{last(j)} < \frac{L_1^{last(j)}+\ell_j}{k} \leq \frac{1+\max\left\{1,\sqrt{\frac{m}{k}}\right\}}{k}$. Finally, since $last(j)$ is the last job assigned to machine j, $L_j(\mathbf{N}) = L_j^{last(j)} + \ell_j \leq L_j^{last(j)} + 1/k$ and the completion time of machine j is $C_j(N_j) = \frac{L_j(\mathbf{N})}{s_j} \leq \frac{L_j^{last(j)}+1/k}{1/k} \leq k\left(\frac{1+\max\left\{1,\sqrt{\frac{m}{k}}\right\}}{k} + \frac{1}{k}\right) = 2 + \max\left\{1, \sqrt{\frac{m}{k}}\right\}$.

The claim follows by choosing $\mathbf{O} = \mathbf{S}$. □

For the case of $k \in (1, 2)$, the following upper bound holds.

Theorem 5. *For the scheduling problem with related machines,* $\mathrm{PoEF}_k \leq \min\left\{n, m, 2k\sqrt{\frac{m}{k-1}}\right\}$ *for any* $k \in (1, 2)$.

We now show that the two upper bounds proved in Theorems 4 and 5 are asymptotically tight.

Proposition 5. *For the scheduling problem with related machines, given any* $k \geq 1$*, there exists an instance for which* $\mathrm{PoEF}_k \geq \max\left\{1, \sqrt{\frac{m}{k}}\right\}$*. Moreover, for any* $k \in \left(1, \frac{3+\sqrt{11}}{6}\right)$*, there exists an instance for which* $\mathrm{PoEF}_k = \Omega\left(\sqrt{\frac{m}{k-1}}\right)$.

Note that, for each $k \in \left[\frac{3+\sqrt{11}}{6}, 2\right)$, it holds $\mathrm{PoEF}_k \leq 2k\sqrt{\frac{m}{k-1}} = O(\sqrt{m})$ by Theorem 5, while, by Proposition 5, we have $\mathrm{PoEF}_k \geq \max\left\{1, \sqrt{\frac{m}{k}}\right\} = \Omega(\sqrt{m})$.

This shows that all the bounds on the PoEF_k presented in this subsection are asymptotically tight.

3.3 Unrelated Machines

In this subsection, we consider the scheduling problem with unrelated machines. In this case we are able to give an exact characterization of the price of k-envy-freeness as witnessed by the upper and lower bounds given in the following.

Theorem 6. *For the scheduling problem with unrelated machines,* $\text{PoEF}_k \leq \left(1 + \frac{1}{k}\right)^{\min\{n,m\}-1}$ *for any* $k \geq 1$.

Proposition 6. *For the scheduling problem with unrelated machines, given any* $k \geq 1$ *and* $\epsilon > 0$*, there exists an instance for which* $\text{PoEF}_k = \left(1 + \frac{1}{k+\epsilon}\right)^{\min\{n,m\}-1}$.

3.4 Complexity

An important feature of the proofs we used to upper bound the PoEF_k in the various cases is that they rely on polynomial time algorithms constructing k-envy-free assignments of reasonable low makespan. In particular, for identical machines with $k \in (1, 2)$, the algorithm used in the proof of Theorem 2 does not require any information to be executed; hence, it indeed constructs a k-envy-free assignment whose performance guarantee coincides with the upper bound on the PoEF_k.

For all the other cases, given an input solution $\mathbf{S}$, all the designed algorithms rearrange the allocations defined by $\mathbf{S}$ so as to obtain in polynomial time a k-envy-free assignment $\mathbf{N}$ such that $\mathcal{M}(\mathbf{N}) \leq \text{PoEF}_k \cdot \mathcal{M}(\mathbf{S})$. This means that, when given as input a solution $\mathbf{S}$ such that $\mathcal{M}(\mathbf{S}) \leq \alpha \cdot \mathcal{M}(\mathbf{O})$, each algorithm computes in polynomial time a k-envy-free assignment $\mathbf{N}$ such that $\mathcal{M}(\mathbf{N}) \leq \alpha \cdot \text{PoEF}_k \cdot \mathcal{M}(\mathbf{O})$. By recalling that there exists a PTAS for the scheduling problem with related and identical machines and a 2-approximation algorithm for the case of unrelated ones and by the fact that our upper bounds of PoEF_k are tight or asymptotically tight, it follows that we are able to compute in polynomial time k-envy-free assignments of best possible quality, when dealing with related and identical machines, and of at least half the best possible quality, when dealing with unrelated ones.

4 Extensions

4.1 Restricted Scheduling

In this subsection, we focus on the case in which a job cannot be assigned to every machine: for any job i there is a set $M_i \subseteq \{1, \ldots, m\}$ containing the machines being admissible for job i. We have to clarify the definition of k-envy-freeness in this setting: in assignment $\mathbf{N}$, machine j k-envies machine j' if $C_j(N_j) > kC_j(N_{j'})$ and for each job $i \in N_{j'}$, $j \in M_i$. It can be easily verified that also in the case of restricted scheduling an envy–free solution always exists. In fact, starting from any feasible assignment, an envy–free solution can be obtained as follows: while there exist two machines j and j' such that j k-envies j', assign to machine j also all the jobs of machine j'.

As already remarked in the introduction, the setting of unrelated machines studied in Subsection 3.3 includes the case of restricted (unrelated) machines, as it is possible to assign a very large value to p_{ij} whenever machine $j \notin M_i$, so that neither an optimal solution, nor a k-envy-free one minimizing the makespan can assign a job to a machine not being admissible for it. Therefore, for the restricted case, it remains to analyze the related and identical settings. In the related case, for which the upper bound provided in Theorem 6 clearly holds, it is possible to modify the instance exploited in Proposition 6 so that it becomes a restricted instance of the related setting, and the following theorem holds.

Proposition 7. *For the restricted scheduling problem with related machines, given any* $k \geq 1$ *and* $\epsilon > 0$*, there exists an instance for which* $\mathrm{PoEF}_k = \left(1 + \frac{1}{k+\epsilon}\right)^{\min\{n,m\}-1}$.

Finally, for the case of identical machines, a trivial upper bound equal to $\min\{n, m\}$ holds as any solution (k-envy–free or not) approximates an optimal one by at most $\min\{n, m\}$, and the following lower bound holds.

Proposition 8. *For the restricted scheduling problem with identical machines, given any* $k \geq 1$*, there exists an instance for which* $\mathrm{PoEF}_k = \Omega(\min\{n, m\})$.

4.2 Sum of Completion Times

In this subsection, we extend our study to the case where the objective is that of minimizing the sum of the completion times of all the machines. We refer to such a case as scheduling SUM problem. Formally, given an assignment $\mathbf{N}$ where $C_j(N_j)$ denotes the completion time of machine j under the assignment $\mathbf{N}$, an optimal assignment minimizes the sum $\sum_{j=1}^{m} C_j(N_j)$. We notice that an optimal solution can be trivially determined by assigning each job to the machine providing it the minimum possible processing time.

By exploiting ideas used for the minimum makespan we are able to show that $\mathrm{PoEF}_k = 1$ for related (and then identical) machines, and that $\mathrm{PoEF}_k = \left(1 + \frac{1}{k}\right)^{\min\{n,m\}-1}$ for unrelated machines. All the details will be given in the full version of the paper.

References

1. Anshelevich, E., Dasgupta, A., Kleinberg, J.M., Tardos, E., Wexler, T., Roughgarden, T.: The Price of Stability for Network Design with Fair Cost Allocation. SIAM Journal on Computing 38(4), 1602–1623 (2008)

2. Bulow, J., Roberts, J.: The Simple Economics of Optimal Auctions. The Journal of Political Economy 97(5), 1060–1090 (1989)
3. Brams, S.J., Taylor, A.D.: An Envy-Free Cake Division Protocol. The American Mathematical Monthly 102(1), 9–18 (1995)
4. Brams, S.J., Taylor, A.D.: Fair Division: From Cake-Cutting to Dispute Resolution. Cambridge University Press (1996)
5. Cohen, E., Feldman, M., Fiat, A., Kaplan, H., Olonetsky, S.: Envy-Free Makespan Approximation. SIAM Journal on Computing 41(1), 12–25 (2012)
6. Caragiannis, I., Kaklamanis, C., Kanellopoulos, P., Kyropoulou, M.: The Efficiency of Fair Division. Theory of Computing Systems 50(4), 589–610 (2012)
7. Christodoulou, G., Koutsoupias, E., Vidali, A.: A Lower Bound for Scheduling Mechanisms. Algorithmica 55(4), 729–740 (2009)
8. Dubins, L.E., Spanier, E.H.: How to cut a cake fairly. American Mathematical Monthly 68, 1–17 (1961)
9. Foley, D.: Resource allocation and the public sector. Yale Economics Essays 7, 45–98 (1967)
10. Feldman, M., Fiat, A., Leonardi, S., Sankowski, P.: Revenue maximizing envy-free multi-unit auctions with budgets. In: Proceedings of the ACM Conference on Electronic Commerce (EC), pp. 532–549 (2012)
11. Fiat, A., Levavi, A.: Tight Lower Bounds on Envy-Free Makespan Approximation. In: Goldberg, P.W. (ed.) WINE 2012. LNCS, vol. 7695, pp. 553–558. Springer, Heidelberg (2012)
12. Graham, R.L.: Bounds for Certain Multiprocessing Anomalies. Bell System Technical Journal 45, 1563–1581 (1966)
13. Goldberg, A.V., Hartline, J.D.: Envy-free auctions for digital goods. In: Proceedings of the ACM Conference on Electronic Commerce (EC), pp. 29–35 (2003)
14. Guruswami, V., Hartline, J.D., Karlin, A.R., Kempe, D., Kenyon, C., McSherry, F.: On profit-maximizing envy-free pricing. In: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1164–1173 (2005)
15. Hartline, J., Ieong, S., Mualem, A., Schapira, M., Zohar, A.: Multi-dimensional envy-free scheduling mechanisms. Technical Report 1144, The Hebrew Univ (2008)
16. Hochbaum, D.S., Shmoys, D.B.: Using Dual Approximation Algorithms for Scheduling Problems: Theoretical and Practical Results. Journal of ACM 34(1), 144–162 (1987)
17. Hochbaum, D.S., Shmoys, D.B.: A Polynomial Approximation Scheme for Scheduling on Uniform Processors: Using the Dual Approximation Approach. SIAM Journal on Computing 17(3), 539–551 (1988)
18. Koutsoupias, E., Vidali, A.: A Lower Bound of $1 + \varphi$ for Truthful Scheduling Mechanisms. Algorithmica 66(1), 211–223 (2013)
19. Lenstra, J.K., Shmoys, D.B.: E Tardos. Approximation algorithms for scheduling unrelated parallel machines. Mathematical Programing 46, 259–271 (1990)
20. Myerson, R.B.: Optimal Auction Design. Mathematics of Operations Research 6, 58–73 (1981)
21. Nagura, J.: On the interval containing at least one prime number. Proceedings of the Japan Academy, Series A 28, 177–181 (1952)
22. Nisan, N., Ronen, A.: Algorithmic Mechanism Design. Games and Economic Behavior 35(1-2), 166–196 (2001)
23. Vickrey, W.: Counterspeculation, Auctions, and Competitive Sealed Tenders. Journal of Finance 16, 8–37 (1961)