A lower bound of $1 + \phi$ for truthful scheduling mechanisms*

Elias Koutsoupias**
and Angelina Vidali ***

Department of Informatics, University of Athens

Abstract. We give an improved lower bound for the approximation ratio of truthful mechanisms for the unrelated machines scheduling problem. The mechanism design version of the problem which was proposed and studied in a seminal paper of Nisan and Ronen is at the core of the emerging area of Algorithmic Game Theory. The new lower bound $1 + \phi \approx 2.618$ is a step towards the final resolution of this important problem.

1 Introduction

We study the classical scheduling problem on unrelated machines [15, 21, 16] from the mechanism-design point of view. There are n machines and m tasks each with different execution times on each machine. The objective of the mechanism is to schedule the tasks on the machines to minimize the makespan, i.e. to minimize the time we have to wait until all tasks are executed. In the mechanism-design version of the problem, the machines are selfish players that want to minimize the execution time of the tasks assigned to them. To overcome their "laziness" the mechanism pays them. With the payments, the objective of each player/machine is to minimize the time of its own tasks minus the payment. A loose interpretation of the payments is that they are given to machines as an incentive to tell the truth. A mechanism is called *truthful* when telling the truth is a dominant strategy for each player: for all declarations of the other players, an optimal strategy of the player is to tell the truth. A classical result in mechanism design, the Revelation Principle, states that for every mechanism, in which each player has a dominant strategy, there is a truthful mechanism which achieves the same objective. The Revelation Principle allows us to concentrate on truthful mechanisms (at least for the class of centralized mechanisms).

---

* Supported in part by IST-15964 (AEOLUS) and the Greek GSRT.
** Email: `elias@di.uoa.gr`
*** Email: `avidali@di.uoa.gr`

A central question in the area of Algorithmic Mechanism Design is to determine the best approximation ratio of mechanisms. This question was raised by Nisan and Ronen in their seminal work [23] and remains wide open today. The current work improves the lower bound on the approximation to $1 + \phi \approx 2.618$, where ϕ is the golden ratio.

A lower bound on the approximation ratio can be of either computational or information-theoretic nature. A lower bound is computational when it is based on some assumption about the computational resources of the algorithm, most commonly that the algorithm is polynomial-time. It is of information-theoretic nature when the source of difficulty is not computational but is imposed by the restrictions of the mechanism framework and more specifically by the truthfulness condition. Our lower bound is entirely information-theoretic: No (truthful) mechanism—including exponential and even recursive algorithms—can achieve approximation ratio better than 2.618.

When we consider the approximation ratio of a mechanism, we ignore the payments and care only about the allocation part of the mechanism. A natural question then arises: Which scheduling (allocation) algorithms are part of truthful mechanisms? There is an elegant and seemingly simple characterization of these mechanisms: Monotone Algorithms. The characterizing property of these algorithms, the Monotonicity Property, implies that when we increase the time of some tasks on a specific machine, the machine will not get any new tasks. Similarly when we decrease the time of some tasks of a specific machine the machine can only get more tasks. In a loose sense, the Monotonicity Property is a combination of these two facts and can be expressed very succinctly. Despite its simplicity, we do not know how to take full advantage of the Monotonicity Property and in this work (as in all previous works on this problem) we apply a very restricted variant of it (Lemma 1).

1.1 Related Work

The scheduling problem on unrelated machines is one of the most fundamental scheduling problems [15, 16]. The problem is NP-complete, it can be approximated within a factor of 2, and it is known that no polynomial-time algorithm can have approximation ratio better than 3/2, unless P=NP [21].

Here we study its mechanism-design version and we improve the results of [8], where we (together with George Christodoulou) gave a lower bound of $1+\sqrt{2}$. The current work can be considered a continuation of it as we use similar tools, in particular Lemma 1; this is essentially the only known tool (used also in the seminal work of Nisan and Ronen [23]), and from this perspective it is unavoidable that we use it again here. However, our techniques are completely different and much more sophisticated than the ones used in [8], where we used simple instances of 3 players and 5 tasks. In this work on the other hand, we use arbitrarily many players and tasks and we obtain the bound of $1 + \phi$ only as the limit when the number of players tends to infinity.

Surprisingly, we are not using anything about the geometrical structure of the mechanism, even though it seemed as if the use of a geometric lemma was a crucial part of the proof in [8]. The main connection between the proof of the 2.61 and the 2.41 lower bound is the use of the second part of Lemma 1 which, albeit being a very simple observation seems very powerful.

Nisan and Ronen [23, 24], who introduced the problem and initiated the algorithmic theory of Mechanism Design gave a truthful n-approximate (polynomial-time) algorithm; they also showed that no mechanism (polynomial-time or not) can achieve approximation ratio better than 2. They conjectured that there is no deterministic mechanism with approximation ratio less than n.

Recent work by Lavi and Swamy [20] improves the upper bound for a special case of the same problem—namely when the processing times have only two possible values low or high—and devise a deterministic 2-approximation truthful mechanism.

Archer and Tardos [4] considered the variant of the problem for related machines. In this case, for each machine there is a single value (instead of a vector), its speed. They provided a characterization of all truthful algorithms in this class, in terms of a monotonicity condition. Using this characterization, they showed that there is an optimal algorithm which is truthful (albeit exponential-time). They also gave a polynomial-time randomized 3-approximation mechanism, which was later improved to a 2-approximation, in [2]. This mechanism is truthful in expectation. Andelman, Azar and Sorani [1] gave a 5-approximation deterministic truthful mechanism, in the same framework, which was then improved by Kovacs [18] to 3-approximation deterministic truthful mechanism, while finally the ratio was reduced to 2.8.

For randomized mechanisms, Nisan and Ronen [23] gave a randomized truthful mechanism for two players, that achieves an approximation ratio of 7/4. Recently, Mu'alem and Schapira [22] extended this result and showed a $7n/8$ upper bound. They also proved a lower bound of $2 - \frac{1}{n}$ for any randomized truthful mechanism for n machines.

Very recently Koutsoupias, Christodoulou and Kovacs [12] gave a more general result. They considered the fractional variant of the same problem and showed a lower bound of $2 - \frac{1}{n}$ (which naturally extends to randomized algorithms). They also gave a $\frac{n+1}{2}$ fractional approximation algorithm.

Related is also some work which has been done in the context of combinatorial auctions which is a generalization of the scheduling problem (see for example [3, 6, 7, 10, 5, 11] and the references within). Saks and Yu [25] proved that for convex domains the Monotonicity Property characterizes the class of social choice functions implementable by truthful mechanisms, generalizing results of [14, 19].

2 Problem definition

We recall here the definitions of the scheduling problem, of the concept of mechanisms, as well as some fundamental properties of them (see [8] for more details, references, and proofs).

Definition 1 (The unrelated machines scheduling problem) *The input to the scheduling problem is a nonnegative matrix t of n rows, one for each machine-player, and m columns, one for each task. The entry t_{ij} (of the i-th row and j-th column) is the time it takes for machine i to execute task j. Let t_i denote the times for machine i, which is the vector of the i-th row. The output is an allocation $x = x(t)$, which partitions the tasks into the n machines. We describe the partition using indicator values $x_{ij} \in \{0, 1\}$: $x_{ij} = 1$ iff task j is allocated to machine i. We should allocate each task to exactly one machine, or more formally $\sum_{j=1}^{m} x_{ij} = 1$.*

In the mechanism-design version of the problem we consider direct-revelation mechanisms. That is, we consider mechanisms that work according to the following protocol:

- Each player i declares the values in row t_i, which is known only to player i.
- The mechanism, based on the declared values, decides how to allocate the tasks to the players.
- The mechanism, based on the declared values, and the allocation of the previous step, decides how much to pay each player.

The mechanism consists of two algorithms, an allocation algorithm and a payment algorithm. The cost of a player (machine) is the sum of the times of the tasks allocated to it minus the payment. One way to think of it is as if the players are lazy and don't want to execute tasks, and the mechanism pays them enough to induce them to execute the tasks. On the other hand, the players know both the allocation and the payment algorithm and may have an incentive to lie in the first step. The class of mechanisms for which the players have no incentive to lie are called truthful mechanisms. Here we consider the strictest version of truthfulness which is the class of dominant truthful mechanisms: In these mechanism truth telling is a dominant strategy, i.e., for every possible declaration of the other players, an optimal strategy of a player is to reveal its true values. This restricts significantly the set of possible algorithms.

On the other hand every mechanism can be turned into an equivalent truthful mechanism. This fact, known in the literature as the Revelation Principle, allows as to concentrate only on truthful mechanisms.

Here we care only about the approximation ratio of the allocation part of the mechanisms. So when we refer to the approximation ratio of a mechanism, we mean the approximation ratio of its allocation part. Since payments are of no importance in this consideration, it would be helpful if we could find a necessary and sufficient condition which characterizes which allocations algorithms are ingredients of truthful mechanisms. Fortunately such condition exists:

Definition 2 (Monotonicity Property) *An allocation algorithm is called monotone if it satisfies the following property: for every two sets of tasks t and t' which differ only on some machine i (i.e., on the i-the row) the associated allocations x and x' satisfy $\sum_{j=1}^{m}(x_{ij} - x'_{ij})(t_{ij} - t'_{ij}) \leq 0$, which can be written more succinctly as a dot product:*

$$(x_i - x'_i) \cdot (t_i - t'_i) \leq 0.$$

The Monotonicity Property characterizes the allocation part of truthful mechanisms. The fact that is necessary and sufficient was shown in [23] and [25] respectively. Although this is a complete characterization, it is not easy to use it and we don't know how to take complete advantage of it.

One fundamental open problem is to find a useful characterization of truthful mechanisms for the scheduling problem. For the much more general problem of mechanism design in arbitrary domains, there is simple characterization by Roberts [17]: The only truthful mechanisms are generalized VCG mechanisms [26, 9, 13]. The scheduling problem is at the other end of the spectrum, where the domain is restricted yet general enough to allow for interesting mechanisms.

Not only we lack such a nice characterization as Roberts Characterization for the domain of the scheduling problem, but we also employ a very specific way to apply the Monotonicity Property. In particular, the only known way to take advantage of the Monotonicity Property is the following lemma [8], which will be the main ingredient of our proof. For completeness we also include the proof of this lemma.

Lemma 1. *Let t be a matrix of processing times and let $x = x(t)$ be the allocation produced by a truthful mechanism. Suppose that we change only the tasks of machine i and in such a way that $t'_{ij} > t_{ij}$ when $x_{ij} = 0$, and $t'_{ij} < t_{ij}$ when $x_{ij} = 1$. The mechanism does not change the allocation to machine i, i.e., $x_i(t') = x_i(t)$. (However, it may change the allocation of other machines).*

Moreover for mechanisms with bounded approximation ratio, suppose that there exists a task with ∞ processing time in all machines except machine i. Suppose further that we change the processing time of this task on machine i to some bounded value and the processing times of the remaining tasks on machine i as above. Then again the allocation of machine i is not affected.

Proof. By the Monotonicity Property, we have

$$\sum_{j=1}^{m}(t_{ij} - t'_{ij})(x_{ij}(t) - x_{ij}(t')) \leq 0.$$

If a task can only be processed by machine i then we must have $x_{ij}(t) = x_{ij}(t') = 1$ and consequently the corresponding term is 0.

For a task j with $x_{ij}(t) = 0$ we have $x_{ij}(t) - x'_{ij}(t) \leq 0$ (whichever the allocation $x_{ij}(t') \in \{0, 1\}$ is). Raising the value of such a tasks means $t_{ij} - t'_{ij} \leq 0$. On the other hand if $x_{ij}(t) = 1$ we have $x_{ij}(t) - x'_{ij}(t) \geq 0$. Lowering the value of such a task we get $t_{ij} - t'_{ij} \geq 0$.

In either case the corresponding product satisfies $(x_{ij}(t) - x'_{ij}(t))(t_{ij} - t'_{ij}) \geq 0$. Every term of this sum is nonnegative and consequently the only way to satisfy the inequality is with equality, by setting $x_{ij}(t) = x_{ij}(t')$ for all j.

To simplify the presentation, when we apply Lemma 1, we will increase or decrease only some values of a machine, not all its values. The understanding will be that *the rest of the values increase or decrease appropriately by a tiny amount which we omit* to keep the expressions simple.

3 A lower bound of $1 + \phi$ for $n \to \infty$ machines.

The main result of this work is

Theorem 1. *There is no deterministic mechanism for the scheduling problem with $n \to \infty$ machines with approximation ratio less than $1 + \phi$.*

We shall build the proof of the theorem around the instance

$$\begin{pmatrix} 0 & \infty & \cdots & \infty & \infty & 1 & a & \cdots & a^{n-2} \\ \infty & 0 & \cdots & \infty & \infty & a & a^2 & \cdots & a^{n-1} \\ & & \ddots & & & & & & \\ \infty & \infty & \cdots & 0 & \infty & a^{n-2} & a^{n-1} & \cdots & a^{2n-4} \\ \infty & \infty & \cdots & \infty & 0 & a^{n-1} & a^n & \cdots & a^{2n-3} \end{pmatrix},$$

where $a \geq 1$ is a parameter and ∞ denotes an arbitrarily high value. Eventually, we will set $a = \phi$ when $n \to \infty$. We let however a to be a parameter for clarity and for obtaining better bounds for finite n.

The lower bound will follow from the fact (which we will eventually prove) that every truthful mechanism with approximation ratio less than $1 + a$ must allocate all $n - 1$ rightmost tasks to the first player. The proof of this fact is by induction. However, the induction needs a stronger induction hypothesis which involves instances of the form

$$T(i_1, \ldots, i_k) = \begin{pmatrix} 0 & \infty & \cdots & \infty & a^{i_1} & a^{i_2} & \cdots & a^{i_k} \\ \infty & 0 & \cdots & \infty & a^{i_1+1} & a^{i_2+1} & \cdots & a^{i_k+1} \\ \vdots & & \ddots & & \vdots & & \ddots & \vdots \\ \infty & \infty & \cdots & 0 & a^{i_1+n-1} & a^{i_2+n-1} & \cdots & a^{i_k+n-1} \end{pmatrix},$$

where $0 \leq i_1 < i_2 < \ldots < i_k$ are natural numbers and $k \leq n - 1$. We allow these instances to have additional tasks for which some value is 0, i.e., additional columns with at least one 0 entry in each one. This is only for technical reasons and will play no significant role in the proof (and it definitely does not affect the optimal cost).

We will call the first n tasks *dummy*. Observe that every mechanism with bounded approximation ratio must allocate the i-th dummy task to player i.

Remark 1. Notice that the optimal allocation has cost a^{i_k}. Furthermore, if $i_1, i_2, \ldots, i_k$ are all successive natural numbers, then the optimal allocation is unique and coincides with the diagonal assignment. Otherwise there are more than one allocations with optimal cost. For example the allocations indicated by stars:

$$\begin{pmatrix} 0* & \infty & \infty & \infty & \infty & 1 & a & a^3* \\ \infty & 0* & \infty & \infty & \infty & a & a^2* & a^4 \\ \infty & \infty & 0* & \infty & \infty & a^2* & a^3 & a^5 \\ \infty & \infty & \infty & 0* & \infty & a^3 & a^4 & a^6 \\ \infty & \infty & \infty & \infty & 0* & a^4 & a^5 & a^7 \end{pmatrix}, \quad \begin{pmatrix} 0* & \infty & \infty & \infty & \infty & 1 & a & a^3* \\ \infty & 0* & \infty & \infty & \infty & a & a^2 & a^4 \\ \infty & \infty & 0* & \infty & \infty & a^2 & a^3* & a^5 \\ \infty & \infty & \infty & 0* & \infty & a^3* & a^4 & a^6 \\ \infty & \infty & \infty & \infty & 0* & a^4 & a^5 & a^7 \end{pmatrix}$$

both have the optimal cost a^3.

We will now show the main technical lemma of the proof.

Lemma 2. *Suppose that a truthful mechanism on $T(i_1, \ldots, i_k)$, does not allocate all non-dummy tasks to the first player. Then we can find another instance for which the approximation ratio is at least $1 + a$.*

Proof. Fix a truthful mechanism and suppose that the first player does not get all non-dummy tasks. In the first part, we manipulate the tasks (by changing their values and using the tools from the previous section) in such a way that we obtain an instance with equal or fewer tasks for which in the allocation of the mechanism

- the first player gets no non-dummy task, and
- every other player gets at most one non-dummy task.

In the second part, we show that instances which satisfy the above two conditions, can be changed to obtain an instance with approximation ratio at least $1 + a$.

1st part: Suppose that the first of the above conditions is not satisfied. That is, suppose that the first player gets some non-dummy task. We can then decrease its value (for the first player) to 0. By the Monotonicity Property and in particular by Lemma 1, the same set of tasks will be allocated to the first player, so he still does not get all non-dummy tasks.

Suppose that the second condition is not satisfied, i.e., there is a player in $\{2, \ldots, n\}$ who gets at least two tasks. We can then lower all the non-zero values allocated to this player to 0 except for one. By the Monotonicity Property and in particular by Lemma 1, the same tasks will be allocated to the player. This guarantees that the first player still does not get all non-dummy tasks.

By repeating the above operations, we decrease the number of non-dummy tasks. We will end up with an instance in which the first player gets no non-dummy task and every other player will get at most one non-dummy task. This process will definitely stop when there is only one non-dummy task left.

Notice that the tasks whose value was changed to 0 remain part of the instance but they will play no particular role in the induction. This is exactly the reason for which we allowed $T(i_1, \ldots, i_k)$ to have additional tasks with at least one 0 entry.

2nd part: We can now assume that there is some $T(i_1, \ldots, i_k)$ for which the above two conditions are satisfied, i.e, the mechanism allocates no non-dummy task to the first player and at most one non-dummy task to each of the other players.

The optimal cost is a^{i_k}. Our aim is to find a task which is allocated to some player j with value at least a^{i_k+1}; we will then increase player j's dummy 0 value to a^{i_k}. Then by Lemma 1, player j will get both tasks with total value at least $a^{i_k+1} + a^{i_k}$. If the optimal value is still a^{i_k}, then the approximation ratio is at least $1 + a$. However, when we raise the dummy 0 to a^{i_k} we may increase the optimal value. The crux of the proof is that there is always an allocated value of at least a^{i_k+1} for which this bad case does not occur. To find such a value we consider two cases:

Case 1: The algorithm assigns a task with value at least a^{i_k+1} to one of the last $n - k$ players. This is the easy case, because we increase the dummy 0 value of this player to a^{i_k} and the optimum is not affected. The reason is that we can allocate the non-dummy tasks to the first k players with cost a^{i_k}.

Example 1. Consider the following instance with $n = 5$ and $k = 3$. Suppose that the mechanism has the allocation indicated by the stars.

$$\begin{pmatrix} 0* & \infty & \infty & \infty & \infty & 1 & a & a^3 \\ \infty & 0* & \infty & \infty & \infty & a & a^2 & a^4* \\ \infty & \infty & 0* & \infty & \infty & a^2* & a^3 & a^5 \\ \infty & \infty & \infty & 0* & \infty & a^3 & a^4* & a^6 \\ \infty & \infty & \infty & \infty & 0* & a^4 & a^5 & a^7 \end{pmatrix}$$

Then we can raise the dummy 0 of the 4-th player to a^3. This does not affect the optimum (which is a^3) but raises the cost of the 4-th player to $a^4 + a^3$.

Case 2: The value of all tasks assigned to the last $n - k$ players is at most a^{i_k}. Consequently the indexes i_ℓs are not successive integers (Remark 1). Let q be the length of the last block of successive indexes, i.e., $k - q$ is the maximum index where there is a gap in the i_ℓ's. More precisely, let $k-q$ be the maximum index for which $i_{k-q}+1 < i_{k-q+1}$. Since player 1 gets no non-dummy task, there is a player $p \in \{q+1,\dots,n\}$ such that some of the last q tasks is allocated to p. We raise the dummy 0 value of player p to a^{i_k}.

We have to show two properties: First that the allocated value to p was at least a^{i_k+1} and that the optimum is not affected. Indeed, the first property follows from the fact that $p > q$ (and by the observation that all values of the last q tasks for the players in $\{q+1,\dots,n\}$ are at least a^{i_k+1}). To show that the optimal solution is not affected consider the allocation which assigns

- the ℓ-th from the end non-dummy task to the ℓ-player, for $\ell < p$
- the ℓ-th from the end non-dummy task to the $(\ell+1)$-player, for $\ell \geq p$

Notice that this allocation assigns no non-dummy task to the p-th player, as it should. The p-th player is allocated the dummy task which was raised from 0 to a^{i_k}. Also, since there is a gap at the $k-q$ position, all allocated values are at most a^{i_k}.

Example 2. Consider the following instance with $n=5$, $k=3$, and $q=2$. Suppose that the mechanism has the allocation indicated by the stars.

$$\begin{pmatrix} 0* & \infty & \infty & \infty & \infty & 1 & a^2 & a^3 \\ \infty & 0* & \infty & \infty & \infty & a & a^3* & a^4 \\ \infty & \infty & 0* & \infty & \infty & a^2 & a^4 & a^5* \\ \infty & \infty & \infty & 0* & \infty & a^3* & a^5 & a^6 \\ \infty & \infty & \infty & \infty & 0* & a^4 & a^6 & a^7 \end{pmatrix}$$

Then $p=3$, and we can raise the dummy 0 of the 3-rd player to a^3. This does not affect the optimum (which allocates the a^3 values), but raises the cost of the 4-th player to $a^5+a^3 \geq a^4+a^3$.

With the above lemma, we can easily prove the main result:

Proof (Proof of Theorem 1). Consider the instance

$$\begin{pmatrix} 0 & \infty & \cdots & \infty & \infty & 1 & a & \cdots & a^{n-2} \\ \infty & 0 & \cdots & \infty & \infty & a & a^2 & \cdots & a^{n-1} \\ & & \ddots & & & & & & \\ \infty & \infty & \cdots & 0 & \infty & a^{n-2} & a^{n-1} & \cdots & a^{2n-4} \\ \infty & \infty & \cdots & \infty & 0 & a^{n-1} & a^n & \cdots & a^{2n-3} \end{pmatrix}.$$

By the previous lemma, either the approximation ratio is at least $1+a$ or all non-dummy tasks are allocated to the first player. In the latter case, we raise the dummy 0 of the 1-st player to a^{n-1}. The optimal cost becomes a^n while the cost of the first player is $1 + a + a^2 + \ldots + a^{n-1}$.

The approximation ratio is at least

$$\min\{1 + \frac{1}{a} + \frac{1}{a^2} + \ldots + \frac{1}{a^{n-1}}, a + 1\}.$$

We select a so that

$$1 + \frac{1}{a} + \frac{1}{a^2} + \ldots + \frac{1}{a^{n-1}} = 1 + a. \tag{1}$$

For $n \to \infty$, this gives $\dfrac{1}{1 - \frac{1}{a}} = 1 + a$.

Thus $a^2 = 1 + a$, and the solution to this equation is $a = \phi$. So the approximation ratio of any mechanism is at least $1 + \phi$. For fixed number of players n, the solution of Equation 1 determines a lower bound for the approximation ratio. For small values of n, the approximation ratio is less than $1 + \phi$ but it converges to it rapidly, as shown in Table 1.

Table 1. The lower bound given by Theorem 1 for a small number of machines.

n	2	3	4	5	6	7	8	...	∞
$1+a$	2	2.324	2.465	2.534	2.570	2.590	2.601	...	$1+\phi$

References

1. Nir Andelman, Yossi Azar, and Motti Sorani. Truthful approximation mechanisms for scheduling selfish related machines. In *22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 69–82, 2005.
2. Aaron Archer. *Mechanisms for Discrete Optimization with Rational Agents*. PhD thesis, Cornell University, January 2004.
3. Aaron Archer, Christos H. Papadimitriou, Kunal Talwar, and Éva Tardos. An approximate truthful mechanism for combinatorial auctions with single parameter agents. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 205–214, 2003.
4. Aaron Archer and Éva Tardos. Truthful mechanisms for one-parameter agents. In *42nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 482–491, 2001.
5. Moshe Babaioff, Ron Lavi, and Elan Pavlov. Mechanism design for single-value domains. In *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference (AAAI)*, pages 241–247, 2005.
6. Yair Bartal, Rica Gonen, and Noam Nisan. Incentive compatible multi unit combinatorial auctions. In *Proceedings of the 9th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, pages 72–87, 2003.

7. Patrick Briest, Piotr Krysta, and Berthold Vöcking. Approximation techniques for utilitarian mechanism design. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 39–48, 2005.
8. George Christodoulou, Elias Koutsoupias, and Angelina Vidali. A lower bound for scheduling mechanisms. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1163–1169, 2007.
9. E. Clarke. Multipart pricing of public goods. *Public Choice*, 8:1733, 1971.
10. Shahar Dobzinski, Noam Nisan, and Michael Schapira. Approximation algorithms for combinatorial auctions with complement-free bidders. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 610–618, 2005.
11. Shahar Dobzinski, Noam Nisan, and Michael Schapira. Truthful randomized mechanisms for combinatorial auctions. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 644–652, 2006.
12. Elias Koutsoupias George Christodoulou and Annamaria Kovacs. Mechanism design for fractional scheduling on unrelated machines. In *to appear in ICALP'07*, 2007.
13. T. Groves. Incentives in teams. *Econometrica*, 41:617631, 1973.
14. Hongwei Gui, Rudolf Müller, and Rakesh V. Vohra. Dominant strategy mechanisms with multidimensional types. In *Computing and Markets*, 2005.
15. D.S. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Co. Boston, MA, USA, 1996.
16. Ellis Horowitz and Sartaj Sahni. Exact and approximate algorithms for scheduling nonidentical processors. *J. ACM*, 23(2):317–327, 1976.
17. Roberts Kevin. The characterization of implementable choice rules. *Aggregation and Revelation of Preferences*, pages 321–348, 1979.
18. Annamaria Kovacs. Fast monotone 3-approximation algorithm for scheduling related machines. In *Algorithms - ESA 2005: 13th Annual European Symposium*, pages 616–627, 2005.
19. Ron Lavi, Ahuva Mu'alem, and Noam Nisan. Towards a characterization of truthful combinatorial auctions. In *44th Symposium on Foundations of Computer Science (FOCS)*, pages 574–583, 2003.
20. Ron Lavi and Chaitanya Swamy. Truthful mechanism design for multi-dimensional scheduling via cycle-monotonicity. In *Proceedings 8th ACM Conference on Electronic Commerce (EC)*, 2007.
21. J.K. Lenstra, D.B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46(1):259–271, 1990.
22. Ah'uva Mu'alem and Michael Schapira. Setting lower bounds on truthfulness. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1143–1152, 2007.
23. Noam Nisan and Amir Ronen. Algorithmic mechanism design (extended abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC)*, pages 129–140, 1999.
24. Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.
25. Michael E. Saks and Lan Yu. Weak monotonicity suffices for truthfulness on convex domains. In *Proceedings 6th ACM Conference on Electronic Commerce (EC)*, pages 286–293, 2005.
26. William Vickrey. Counterspeculations, auctions and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.