Scheduling Selfish Tasks: About the Performance of Truthful Algorithms

George Christodoulou[1], Laurent Gourvès[2], and Fanny Pascual[3]

[1] Max-Planck-Institut für Informatik, Saarbrücken, Germany
gchristo@mpi-inf.mpg.de
[2] LAMSADE, CNRS UMR 7024, Université de Paris-Dauphine, Paris, France
laurent.gourves@lamsade.dauphine.fr
[3] Equipe MOAIS (CNRS-INRIA-INPG-UJF), Grenoble, France
fanny.pascual@imag.fr

Abstract. This paper deals with problems which fall into the domain of selfish scheduling: a protocol is in charge of building a schedule for a set of tasks without directly knowing their length. The protocol gets these informations from agents who control the tasks. The aim of each agent is to minimize the completion time of her task while the protocol tries to minimize the maximal completion time. When an agent reports the length of her task, she is aware of what the others bid and also of the protocol's algorithm. Then, an agent can bid a false value in order to optimize her individual objective function. With erroneous information, even the most efficient algorithm may produce unreasonable solutions. An algorithm is truthful if it prevents the selfish agents from lying about the length of their task. The central question in this paper is: *"How efficient a truthful algorithm can be?* We study the problem of scheduling selfish tasks on parallel identical machines. This question has been raised by Christodoulou et al [8] in a distributed system, but it is also relevant in centrally controlled systems. Without considering side payments, our goal is to give a picture of the performance under the condition of truthfulness.

Keywords: scheduling, algorithmic game theory, truthful algorithms.

1 Introduction

The Internet is a complex distributed system involving many autonomous entities (*agents*). Protocols organize this network, using the data held by these agents and trying to maximize the social welfare. Agents are often supposed to be trustworthy but this assumption is unrealistic in some settings as they might try to manipulate the protocol by reporting false information in order to maximize their own profit. With false information, even the most efficient protocol may lead to unreasonable solutions if it is not designed to cope with the selfish behavior of the single entities. Then, it is natural to ask the following question: *How efficient a protocol can be if it guarantees that no agent has incentive to lie?*

G. Lin (Ed.): COCOON 2007, LNCS 4598, pp. 187–197, 2007.

In this paper, we deal with the problem of scheduling n selfish tasks on m identical parallel machines. We consider two distinct settings in which the aim is to minimize the *makespan*, i.e. the maximum completion time. The first setting is centralized, while the second one is distributed. Both problems share the following characteristics. Each task is owned by an agent[1]. The length l_i of a task i is known to its owner only. The agents, considered as players of a non-cooperative game, want to minimize the completion time of their tasks. The protocol builds the schedule with rules known to all players and fixed in advance. In particular, mixing the execution of two jobs (like round-robin) is not allowed. Before the execution begins, the agents report a value representing the length of their tasks. We assume that every agent behave rationally and selfishly. Each one is aware of the situation the others face and tries to optimize her own objective function. Thus an agent can report a value which is not equal to her real length. Practically, an agent can add "fake" data to artificially increase the length of her task if it decreases her completion time. This selfish behavior can prevent the protocol to produce a reasonable (i.e. close to the social welfare) schedule. Without considering side payments, which are often used with the aim of inciting the agents to report their real value, some algorithmic tools can simultaneously offer a guarantee on the quality of the schedule (its makespan is not arbitrarily far from the optimum) and guarantee that the solution is *truthful* (no agent can lie and improve her own completion time). For both centralized and distributed settings, our goal is to give lower and upper bounds on the performance under the condition of truthfulness. It is important to mention that we do not strictly restrict the study to polynomial time algorithms.

Since the length of a task is private, each agent bids a value which represents the length of her task. We assume that an agent cannot shrink the length of her task (otherwise she will not get her result), but if she can decrease her completion time by bidding a value larger than the real one, then she will do so. We also assume that an agent does not report a distribution on different lengths. A player may play according to a distribution, but she just announces the outcome, so the protocol does not know if she lies.

In the *centralized setting*, the strategy of agent i is a value b_i representing the length of her task. The protocol, called an algorithm, is in charge of indicating when and on which machine a task will be scheduled. An algorithm is *truthful* when no agent has incentive to report a false value. We focus on the performance of truthful algorithms with respect to the makespan of the schedule. In particular, we are interested in giving lower and upper bounds on the *approximation ratio* that a (deterministic or randomized) truthful algorithm can achieve. For example, a truthful algorithm can be obtained by greedily scheduling the tasks following the increasing order of their lengths. This algorithm, known as SPT, produces a $(2-1/m)$-approximate schedule [11]. *Are there truthful algorithms with better approximation guarantee for the considered scheduling problem?*

[1] We equally refer to a task and its owner since we assume that two tasks cannot be held by the same agent.

In the *distributed setting*, the strategy of agent i is a couple (M_i, b_i), where M_i is the machine which will execute the task and b_i is the length bidden. As opposed to the centralized setting, the agents choose their machine and M_i can be a probability distribution on different machines. The protocol, called a *coordination mechanism* in this context [8], consists in selecting a *scheduling policy* for each machine (e.g. scheduling the tasks in order of decreasing lengths). An important and natural condition is due to the decentralized nature of the problem: the scheduling on a machine should depend only on the tasks assigned to it, and should be independent of the tasks assigned to the other machines. A coordination mechanism is *truthful* when no agent has incentive to lie on the length of her task. Using the *price of anarchy* [14], we study the performance of truthful coordination mechanisms with respect to the makespan. The price of anarchy of a coordination mechanism is, in the context, equal to the largest ratio between the makespan of a schedule where agent's strategies form a *Nash equilibrium*[2] and the optimal makespan.

Interestingly, it is possible to slightly transform the SPT algorithm in a truthful coordination mechanism, as suggested in [8]: each machine P_j schedules its tasks in order of increasing lengths, and adds at the very beginning of the schedule a small delay equal to $(j-1)\varepsilon$ times the length of the first task. By this way, and if ε is small enough, the schedule obtained in a Nash equilibrium is similar to the one returned by the SPT algorithm (excepted the small delays at the beginning of the schedule). When ε is negligible, the price of anarchy of this coordination mechanism is $2-1/m$. *Are there truthful coordination mechanisms with better price of anarchy for the considered scheduling problem?*

For both centralized algorithms and coordination mechanisms, we consider the two following execution models:

- **Strong model of execution:** If the owner of task i bids $b_i \geq l_i$, then the execution time will still be l_i (i.e. the task will be completed l_i time units after its start).
- **Weak model of execution:** If the owner of task i bids $b_i \geq l_i$, then the execution time will be b_i (i.e. the task will be completed b_i time units after its start).

The strong execution model corresponds to the case where tasks have to be linearly executed – from their beginning to their end–, whereas the weak execution model corresponds to the case where a task can be executed in any order[3] (and the "fake" part of the task is not anymore necessarily executed at the end), or when the machine returns the result of the task only at the end of its execution. Depending on the applications of the scheduling problem, either the strong or the weak model of execution will be used.

[2] Situation in which no agent can unilaterally change her strategy and improve her own completion time. A Nash equilibrium is *pure* if each agent has a pure strategy : each agent chooses only one machine. A Nash equilibrium is *mixed* if the agents give a probability distribution on the machines on which they will go.

[3] Nevertheless, the execution of two jobs is never interlaced.

Related Work

The field of *Mechanism Design* can be useful to deal with the selfishness of the agents. Its main idea is to pay the agents to convince them to perform strategies that help the system to optimize a global objective function. The most famous technique for designing truthful mechanisms is perhaps the Vickrey-Clarke-Groves (VCG) mechanism [20,7,12]. However, when applied to combinatorial optimization problems, this mechanism guarantees the truthfulness under the hypothesis that the objective function is *utilitarian* (i.e. the value of the objective function is equal to the sum of the agents individual objective functions) and that the mechanism is able to compute the optimum. Archer and Tardos introduce in [4] a method which allows to design truthful mechanisms for several combinatorial optimization problems to which the VCG mechanism does not apply. However, both approaches cannot be applied to our problem.

Scheduling selfish agents has been intensively studied these last years, started with the seminal work of Nisan and Ronen [17], and followed by a series of papers [1,2,4,6,9,15,16]. However, all these works differ from ours since in their case, the selfish agents are the machines while here we consider that the agents are the tasks. Furthermore, they use side payments whereas we focus on truthful algorithms without side payments.

A more closely related work is the one of Christodoulou et al [8] who considered the same model but only in the distributed context of coordination mechanisms. They proposed different coordination mechanisms with a price of anarchy better than the one of the SPT coordination mechanism. Nevertheless, these mechanisms are not truthful. In [13], the authors gave coordination mechanisms for the same model for related machines (i.e. machines can have different speeds), but their mechanisms are also not truthful.

In [3], the authors gave a truthful randomized algorithm for the strong model of execution defined before, and they gave, for the weak model of execution, a coordination mechanism which is truthful if there are two machines and if the lengths of the tasks are powers of a certain constant. An optimal (but exponential time) truthful randomized algorithm and a truthful randomized PTAS for the weak model of execution appear in [18,19]. The technique consists in computing an optimal (resp. a $(1+\varepsilon)$-approximate) schedule and each machine executes its tasks in a random order (the truthfulness is due to the introduction of fictitious tasks which guarantee that all the machines have the same load).

Another related work is the one of Auletta et al. who considered in [5] the problem of scheduling selfish tasks in a centralized case. Their work differs from ours since they considered that each machine uses a round and robin policy and thus that the completion of each task is the completion time of the machine on which the task is (this model is known as the KP model). They considered that the tasks can lie in both directions, and that there are some payments.

Contribution and Organization of the Article

Sections 3 and 4 are devoted to the centralized setting. In particular, we study the strong (resp. weak) model of execution in Section 3 (resp. Section 4). Results on the distributed setting are presented in Section 5 for both execution models.

Table 1 gives a summary of the bounds that we are aware of (those with a † are presented in this article). LB stands for "Lower bound", UB for "Upper bound" and NE for "Nash equilibria". Due to space constraints, some proofs are omitted.

Table 1. Bounds for m identical machines

Strong model of execution:

	Deterministic		Randomized	
	LB	UB	LB	UB
centralized setting	$2-\frac{1}{m}$ †	$2-\frac{1}{m}$ [8]	$\frac{3}{2}-\frac{1}{2\,m}$ †	$2-\frac{1}{m+1}\left(\frac{5}{3}+\frac{1}{3m}\right)$ [3]
distributed setting	$2-\frac{1}{m}$ (pure NE) † $\frac{3}{2}-\frac{1}{2\,m}$ (mixed NE) †	$2-\frac{1}{m}$ [8]	$\frac{3}{2}-\frac{1}{2\,m}$ †	$2-\frac{1}{m}$

Weak model of execution:

	Deterministic		Randomized	
	LB	UB	LB	UB
centralized setting	$m=2: 1+\frac{\sqrt{105}-9}{12}>1.1$ † $m\geq 3: \frac{7}{6}>1.16$ †	$\frac{4}{3}-\frac{1}{3\,m}$ †	1 [18,19]	1 [18,19]
distributed setting	$\frac{1+\sqrt{17}}{4}>1.28$ (pure NE) †	$2-\frac{1}{m}$	$1+\frac{\sqrt{13}-3}{4}>1.15$ † (pure NE)	$2-\frac{1}{m}$

2 Notations

We are given m machines (or processors) $\{P_1,\ldots,P_m\}$, and n tasks $\{1,\ldots,n\}$. Let l_i denote the real execution time (or length) of task i. We use the identification numbers to compare tasks of the same (bidden) lengths: we will say that task i, which bids b_i, is larger than task j, which bids b_j, if and only if $b_i > b_j$ or ($b_i = b_j$ and $i > j$). It is important to mention that an agent cannot lie on her (unique) identification number.

A randomized algorithm can be seen as a probability distribution over deterministic algorithms. We say that a (randomized) algorithm is truthful if for every task the expected completion time when she declares her true length is smaller than or equal to her expected completion time in the case where she declares a larger value. More formally, we say that an algorithm is *truthful* if $E_i[l_i] \leq E_i[b_i]$, for every i and $b_i \geq l_i$, where $E_i[b_i]$ is the expected completion time of task T_i if she declares b_i. In order to evaluate the quality of a randomized algorithm, we use the notion of expected approximation ratio.

We will refer in the sequel to the list scheduling algorithms LPT and SPT, where LPT (resp. SPT) [11] is the algorithm which greedily schedules the tasks, sorted in order of decreasing (resp. increasing) lengths: this algorithm schedules, as soon as a machine is available, the largest (resp. smallest) task which has not yet been scheduled. An LPT (resp. SPT) schedule is a schedule returned by the LPT (resp. SPT) algorithm.

3 About Truthful Algorithms for the Strong Model of Execution

3.1 Deterministic Algorithms

We saw that the deterministic algorithm SPT, which is $(2-\frac{1}{m})$-approximate, is truthful. Let us now show that there is no truthful deterministic algorithm with a better approximation ratio.

Theorem 1. *Let us consider that we have m identical machines. There is no truthful deterministic algorithm with an approximation ratio smaller than $2-\frac{1}{m}$.*

Proof. Let us suppose that we have $n = m\,(m-1)+1$ tasks of length 1. Let us suppose that we have a truthful deterministic algorithm $\mathcal{A}$ which has an approximation ratio smaller than $(2-1/m)$ Let t be the task which has the maximum completion time, C_t, in the schedule returned by $\mathcal{A}$. We know that $C_t \geq m$.

Let us now suppose that task t bids m instead of 1. We will show that the completion time of t is then smaller than m. Let OPT be the makespan of an optimal solution where there are $n-1 = m\,(m-1)$ tasks of length 1 and a task of length m. We have: $OPT = m$. Since te approximation ratio of algorithm $\mathcal{A}$ is smaller than $(2-1/m)$, the makespan of the schedule it builds with this instance is smaller than $(2-1/m)\,m = 2\,m-1$. Thus, the task of length m starts before time $(m-1)$. Thus, if task t bids m instead of 1, it will start before time $m-1$ and be completed one time unit after, that is before time m. Thus task t will decrease its completion time by bidding m instead of 1, and algorithm $\mathcal{A}$ is not truthful.

Note that we can generalize this result to the case of related machines: we have m machines $P_1, \ldots, P_m$, such that machine P_i has a speed v_i, $v_1 = 1$, and $v_1 \leq \ldots \leq v_m$. By this way, the bound becomes $2-\frac{v_m}{\sum_{i=1}^{m} v_i}$.

Concerning the strong model of execution, no deterministic algorithm can outperform SPT in the centralized setting. Then, it is interesting to consider randomized algorithms to achieve a better approximation ratio.

3.2 Randomized Algorithms

In [3], the authors present a randomized algorithm which consists in returning a LPT schedule with a probability $1/(m+1)$ and a slightly modified SPT schedule with a probability $m/(m+1)$. They obtain a truthful algorithm whose expected approximation ratio improves $2-\frac{1}{m}$ but no instance showing the tightness of their analysis is provided. A good candidate should be simultaneously a tight example for both LPT and SPT schedules. We are not aware of the existence of such an instance and we believe in a future improvement of this upper bound. The following Theorem provides a lower bound.

Theorem 2. *Let us consider that we have m identical machines. There is no truthful randomized algorithm with an approximation ratio smaller than $\frac{3}{2}-\frac{1}{2m}$.*

Generalizing this result to the case of related machines, the bound becomes $\frac{3}{2} - \frac{v_m}{2\sum_{i=1}^{m} v_i}$.

4 About Truthful Algorithms for the Weak Model of Execution

4.1 A Truthful Deterministic Algorithm

We saw in the Section 3 that SPT is a truthful and $(2 - 1/m)$-approximate algorithm for the strong model of execution, and that no truthful deterministic algorithm can have a better approximation ratio. If we consider the weak model of execution, we can design a truthful deterministic algorithm, called LPT_{mirror}, with a better performance guarantee. We are given n tasks $\{1, \ldots, n\}$ which bid lengths $b_1, \ldots, b_n$. Make a schedule σ_{LPT} with the LPT list algorithm. Let C_{max}^{OPT} be the optimal makespan. Let $p(i)$ be the machine on which the task i is executed in σ_{LPT}. Let C_i be date at which the task i ends in σ_{LPT}. LPT_{mirror} returns the schedule in which task i is executed on machine $p(i)$ and starts at time $\big(4/3 - 1/(3m)\big)C_{max}^{OPT} - C_i$.

Theorem 3. *LPT_{mirror} is a deterministic, truthful and $(\frac{4}{3} - \frac{1}{3m})$-approximate algorithm.*

Proof. We are given n tasks with true lengths $l_1, \ldots, l_n$. Let us suppose than each task has bidden a value, and that task i bids $b_i > l_i$. This can make the task i start earlier in σ_{LPT} but never later. In addition, the optimal makespan when i bids $b_i > l_i$ is necessarily larger than or or equal to the optimal makespan when task i reports its true length.

Let S_i be the date at which task i starts to be executed in σ_{LPT}. The completion time of task i in LPT_{mirror} is $\big(4/3 - 1/(3\,m)\big)OPT - C_i + b_i = \big(4/3 - 1/(3\,m)\big)OPT - S_i$ because $S_i = C_i - b_i$. By bidding $b_i > l_i$, task i can only increase its completion time in the schedule returned by LPT_{mirror} because OPT does not decrease and S_i does not increase. Thus task i does not have incentive to lie.

Since the approximation ratio of the schedule obtained with the LPT list algorithm is at most $\big(4/3 - 1/(3m)\big)$ [11], the schedule returned by LPT_{mirror} is clearly feasible and its makespan is, by construction, $\big(4/3 - 1/(3m)\big)$-approximate. Thus LPT_{mirror} is a truthful and $(\frac{4}{3} - \frac{1}{3m})$-approximate algorithm.

Note that LPT_{mirror} is not a polynomial time algorithm, since we need to know the value of the makespan in an optimal solution, which is an NP-hard problem [10]. However, it is possible to have a polynomial time algorithm which is $\big(4/3 - 1/(3m)\big)$-approximate, even if some tasks do not bid their true values. Consider the following simple algorithm: we first compute a schedule σ_{LPT} with the LPT algorithm. Let $p(i)$ be the machine on which the task i is executed in σ_{LPT}, let C_i be the completion time of task i in σ_{LPT}, and let C_{max} be the makespan of

σ_{LPT}. We then compute the final schedule σ' in which task i is scheduled on $p(i)$ and starts at time $C_{max} - C_i$.

We can show that this algorithm is $\big(4/3 - 1/(3m)\big)$-approximate (i.e. the schedule returned by this algorithm is at most $\big(4/3 - 1/(3m)\big)$ times larger than the optimal schedule in which all the tasks bid their true values). We can show this by the following way. We suppose that all the tasks except i have bidden some values. Let $\sigma_{LPT}(b_i)$ be the schedule σ_{LPT} obtained when i bids b_i, let $S_i(b_i)$ be the date at which task i starts to be executed in $\sigma_{LPT}(b_i)$, and let $C_{max}(\sigma_{LPT}(b_i))$ be the makespan of $\sigma_{LPT}(b_i)$. The completion time of task i (which bids b_i) in σ' is equal to $C_{max}(\sigma_{LPT}(b_i)) - S_i(b_i)$. Since with the LPT algorithm, tasks are scheduled in decreasing order of lengths, if $b_i > l_i$ then $S_i(b_i) \leq S_i(l_i)$. Thus, whatever the values bidden by the other tasks are, i has incentive to lie and bid $b_i > l_i$ only if $C_{max}(\sigma_{LPT}(b_i)) < C_{max}(\sigma_{LPT}(l_i))$. Since this is true for each task, no task will unilaterally lie unless this decreases the makespan of the schedule. The makespan of the schedule σ' in which all the tasks bid their true values is $\big(4/3 - 1/(3m)\big)$-approximate, and then the solution returned by this algorithm will also be $\big(4/3 - 1/(3m)\big)$-approximate.

4.2 Deterministic Algorithms: Lower Bounds

We suppose that the solution returned by an algorithm depends on the length and the identification number of each task, even those which can be identified with their unique length.

Theorem 4. *Let us consider that we have two identical machines. There is no truthful deterministic algorithm with an approximation ratio smaller than* $1 + (\sqrt{105} - 9)/12 \approx 1.1039$.

Theorem 5. *Let us consider that we have* $m \geq 3$ *identical machines. There is no truthful deterministic algorithm with an approximation ratio smaller than* $7/6$.

The assumption made to derive Theorems 4 and 5 is, in a sense, stronger than the usual one since we suppose that the solution returned by an algorithm for two similar instances (same number of tasks, same lengths but different identification numbers) can be completely different. If we relax this assumption, i.e. if identification numbers are only required for the tasks which have the same length, the bound presented in Theorem 4 can be improved to $7/6$.

Theorem 6. *Let us consider that we have two identical machines. No truthful deterministic algorithm can be better than* $7/6$*-approximate if it does not take into account the identification number of tasks whose length is unique.*

5 About Truthful Coordination Mechanisms

Let $\rho \geq 1$. If there is no truthful deterministic algorithm which has an approximation ratio of ρ, then there is no truthful deterministic coordination mechanism which always induce pure Nash equilibria and which has a price of anarchy

smaller than or equal to ρ. Indeed, if this was not the case, then the deterministic algorithm which consists in building the schedule obtained in a pure Nash equilibrium with this ρ-approximate coordination mechanism would be a ρ-approximate truthful deterministic algorithm.

Likewise, if there is no truthful (randomized) algorithm which has an approximation ratio of ρ, then there is no truthful coordination mechanism which has a price of anarchy smaller than or equal to ρ. Indeed, if this was not the case, the algorithm which consists in building the schedule obtained in a Nash equilibrium with this ρ-approximate coordination mechanism would be a ρ-approximate truthful algorithm.

This observation leads us to the following results for the strong model of execution. We deduce from Theorem 1 that there is no truthful deterministic coordination mechanism which always induce pure Nash equilibria and which has a price of anarchy smaller than $2 - 1/m$. Thus there is no truthful coordination mechanism which performs better than the truthful SPT coordination mechanism, whose price of anarchy tends towards $2 - 1/m$. We deduce from Theorem 2 that there is no truthful coordination mechanism which has a price of anarchy smaller than $\frac{3}{2} - \frac{1}{2\,m}$. We now consider the weak model of execution.

Theorem 7. *If we consider the weak model of execution, there is no truthful deterministic coordination mechanism which induces pure Nash equilibria, and which has a price of anarchy smaller than* $\frac{1+\sqrt{17}}{4} \approx 1.28$.

Proof. Let us first prove this result in the case where there are two machines, P_1, and P_2. Let $\varepsilon > 0$. Let us suppose that there exists a truthful coordination mechanism $\mathcal{M}$ with a price of anarchy of $\frac{1+\sqrt{17}}{4} - \varepsilon$. Let us consider the following instance I_1: three tasks of length 1. Since $\mathcal{M}$ is a deterministic coordination mechanism which induces pure Nash equilibria, there is at least a task in I_1 which has a completion time larger than or equal to 2. Let t be such a task.

Let us first consider this instance I_2: we have two tasks of length $\frac{-1+\sqrt{17}}{2} \approx 1.56$. Since $\mathcal{M}$ is $(\frac{1+\sqrt{17}}{4} - \varepsilon)$-approximate, there is one task on each machine, and each task is completed before time $\frac{-1+\sqrt{17}}{2} \times \frac{1+\sqrt{17}}{4} = 2$. Thus, when it has a task of length $\frac{-1+\sqrt{17}}{2}$, each machine must end it before time 2.

Let us now consider the following instance I_3: two tasks of length 1, and a task of length $\frac{-1+\sqrt{17}}{2}$. Since $\mathcal{M}$ is $(\frac{1+\sqrt{17}}{4} - \varepsilon)$-approximate, the task of length $\frac{-1+\sqrt{17}}{2}$ is necessarily alone on its machine (without loss of generality, on P_2). As we have seen it, P_2 must schedule this task before time 2. Thus, task t of instance I_1, has incentive to bid $\frac{-1+\sqrt{17}}{2}$ instead of 1: by this way it will end before time 2, instead of a time larger than or equal to 2.

We can easily extend this proof in the case where there are more than 2 machines, by having $m + 1$ tasks of length 1 in I_1; m tasks of length $\frac{-1+\sqrt{17}}{2}$ in I_2; and m tasks of length 1 and a task of length $\frac{-1+\sqrt{17}}{2}$ in I_3.

Theorem 8. *If we consider the weak model of execution, there is no truthful coordination mechanism which induces pure Nash equilibria, and which has a price of anarchy smaller than* $1 + \frac{\sqrt{13}-3}{4} \approx 1.15$.

6 Conclusion

We showed that, in the strong model of execution, the list algorithm SPT, which has an approximation ratio of $2 - 1/m$ is the best truthful deterministic algorithm, and that there is no truthful randomized algorithm which has an approximation ratio smaller than $3/2 - 1/(2\,m)$. On the contrary, if we relax the constraints on the execution model, i.e. if the result of a task which bid b is given to this task only b time units after its start, then we can obtain better results. In this model of execution, there is a truthful $4/3 - 1/(3\,m)$-approximate deterministic algorithm and a truthful optimal randomized algorithm. For both execution models, we also gave lower bounds on the approximation ratios that a truthful coordination mechanism can have.

As a future work, it would be interesting to improve the results for which a gap between the lower and the upper bound exists. For example, we believe that the lower bound $\frac{1+\sqrt{17}}{4}$ (lower bound on the performance of a truthful deterministic coordination mechanism for the weak model of execution) can be improved to $3/2$ for two machines.

Another direction would be to restrict the study to truthful algorithms (or coordination mechanisms) which run in polynomial time. Giving improved lower bounds which rely on a computational complexity argument would be very interesting.

Acknowledgments. We thank Elias Koutsoupias for helpful suggestions and discussions on the problem.

References

1. Ambrosio, P., Auletta, V.: Deterministic Monotone Algorithms for Scheduling on related Machines. In: Persiano, G., Solis-Oba, R. (eds.) WAOA 2004. LNCS, vol. 3351, pp. 267–280. Springer, Heidelberg (2005)
2. Andelman, N., Azar, Y., Sorani, M.: Truthful Approximation Mechanisms for Scheduling Selfish Related Machines. In: Diekert, V., Durand, B. (eds.) STACS 2005. LNCS, vol. 3404, pp. 69–82. Springer, Heidelberg (2005)
3. Angel, E., Bampis, E., Pascual, F.: Truthful Algorithms for Scheduling Selfish Tasks on Parallel Machines. In: Deng, X., Ye, Y. (eds.) WINE 2005. LNCS, vol. 3828, pp. 698–707. Springer, Heidelberg (2005)
4. Archer, A., Tardos, E.: Truthful Mechanisms for One-Parameter Agents. In: Proc. of FOCS 2001, pp. 482-491 (2001)
5. Auletta, V., Penna, P., De Prisco, R., Persiano, P.: How to Route and Tax Selfish Unsplittable Traffic. In: Proc. of SPAA 2004, pp. 196-204 (2004)

6. Auletta, V., De Prisco, R., Penna, P., Persiano, P.: Deterministic Truthful Approximation Mechanisms for Scheduling Related Machines. In: Diekert, V., Habib, M. (eds.) STACS 2004. LNCS, vol. 2996, pp. 608–619. Springer, Heidelberg (2004)
7. Clarke, E.: Multipart pricing of public goods. Public Choices, pp. 17-33 (1971)
8. Christodoulou, G., Koutsoupias, E., Nanavati, A.: Coordination Mechanisms. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 345–357. Springer, Heidelberg (2004)
9. Christodoulou, G., Koutsoupias, E., Vidali, A.: A lower bound for scheduling mechanisms. In: Proc. of SODA 2007 (2007)
10. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co (1979)
11. Graham, R.: Bounds on multiprocessor timing anomalies. In: SIAM Jr. on Appl. Math. vol. 17(2), pp. 416-429 (1969)
12. Groves, T.: Incentive in teams. Econometrica 41(4), 617–631 (1973)
13. Immorlica, N., Li, L., Mirrokni, V.S., Schulz, A.: Coordination Mechanisms for Selfish Scheduling. In: Deng, X., Ye, Y. (eds.) WINE 2005. LNCS, vol. 3828, pp. 55–69. Springer, Heidelberg (2005)
14. Koutsoupias, E., Papadimitriou, C.: Worst Case Equilibria. In: Meinel, C., Tison, S. (eds.) STACS 99. LNCS, vol. 1563, pp. 404–413. Springer, Heidelberg (1999)
15. Kovács, A.: Fast monotone 3-approximation algorithm for scheduling related machines. In: Brodal, G.S., Leonardi, S. (eds.) ESA 2005. LNCS, vol. 3669, pp. 616–627. Springer, Heidelberg (2005)
16. Mu'alem, A., Schapira, M.: Setting lower bounds on truhfulness. In: Proc. of SODA 2007 (2007)
17. Nisan, N., Ronen, A.: Algorithmic mechanism design. In: Proc. STOC 1999, pp. 129-140 (1999)
18. Pascual, F.: Optimisation dans les réseaux : de l'approximation polynomiale à la théorie des jeux. Ph.D Thesis, University of Evry, France, 2006 (in french).
19. Tchetgnia, A-A.: Truthful algorithms for some scheduling problems. Master Thesis MPRI, École Polytechnique, France (2006)
20. Vickrey, W.: Counterspeculation, auctions and competitive sealed tenders. J. Finance 16, 8–37 (1961)