Brief Announcement: On the Truthfulness and the Approximation for Scheduling Selfish Tasks

Eric Angel
IBISC, Univ. of Evry, France
e.angel@ibisc.fr

Evripidis Bampis
IBISC, Univ. of Evry, France
bampis@ibisc.fr

Fanny Pascual
LIG-IMAG, Grenoble, France
fanny.pascual@imag.fr

Alex-Ariel Tchetgnia
IBISC, Univ. of Evry, France
tchetgnia@gmail.com

Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

General Terms

Algorithms

Keywords

Scheduling, Approximation Algorithms, Truthfulness, Game Theory

1. INTRODUCTION

We consider the problem of designing truthful mechanisms for scheduling n tasks on a set of m parallel (identical or related) machines in order to minimize the *makespan*. Each task is owned by a selfish agent whose aim is to minimize the completion time of her task. We focus on the following process: at first the agents declare the length of their tasks, then given these bids the scheduler allocates the tasks to the machines. The objective of the scheduler is to minimize the *makespan*, i.e. the date at which the last task finishes its execution, while the aim of each agent is to minimize its *completion time* and thus an agent may lie if by doing so, her task can finish earlier.

More formally, we are given m machines $M_1, M_2, \ldots, M_m$ with speeds $s_1, s_2, \ldots, s_m$, respectively. We assume that $s_1 \leq s_2 \leq \ldots \leq s_m$ and we denote by $r = s_m/s_1$ the ratio between the largest and smallest speeds. We are also given a set of tasks $H = \{T_1, \ldots, T_n\}$. For every $i \leq n$, we denote by $l(T_i)$, or sometimes l_i, the length of task T_i and $C(T_i)$ its completion time. If task T_i is scheduled on machine M_j, its processing time is $l(T_i)/s_j$. We will say that a task T_i is *larger* than a task T_j if and only if $l(T_i) > l(T_j)$ or ($l(T_i) = l(T_j)$ and $i > j$).

Our goal is to design scheduling algorithms which minimize the *makespan* ($\max_j\{C(T_j)\}$) and that simultaneously prevent the agents from reporting false values about their length. The problem is denoted by $Q||C_{\max}$ for the case with different speeds, and by $P||C_{\max}$ when the machines are identical.

SPAA'07, June 9–11, 2007, San Diego, California, USA.
ACM 978-1-59593-667-7/07/0006.

1.1 Models

There are different models depending on the values that a task may report and its resulting length in the returned schedule. More formally, let b_i be the length that the task T_i reports. In the *strong model of execution*, if a task T_i bids a value $b_i \neq l(T_i)$, then the length of that task in the schedule remains $l(T_i)$. There are two variants of the strong model: the *w-strong* (without shrink) model in which no task may declare a value smaller than its real length, and the *g-strong* (general) model where the tasks may report any value. In the *weak model of execution*, if a task T_i bids a value b_i, its length in the schedule will be $\max\{l(T_i), b_i\}$ (in order that task T_i has enough time to be executed). In the *w-weak* model of execution no task may declare a value smaller than its real length, whereas in the *g-weak* model a task may report any arbitrary value.

1.2 Type of truthful algorithms

We consider that the agents are selfish but rational, i.e. they declare a false value only if they can reduce (strictly) their individual completion times. We will refer to two types of algorithms: *truthful* algorithms which guarantee that no agent has incentive to report a false value regardless of what is declared by the other tasks, and *equilibria-truthful* algorithms which guarantee that none of the tasks has incentive to report a false value given that all the other tasks have reported their true values.

1.3 Related works

Auletta et al. [2] consider the $Q||C_{max}$ problem in which the individual function of each task is the *completion time of the machine* on which it is executed, while the global objective function is the *makespan*. They consider the *g-strong* model of execution and they provide *equilibria-truthful mechanisms* that **use payments**. In [1], the authors consider the $P||C_{max}$ problem in which the individual objective function of each task is *its* completion time. They consider the *w-strong* model of execution and they propose a randomized algorithm that is proved to be truthful and with an (expected) approximation ratio of $2 - \frac{1}{m+1}(\frac{5}{3} + \frac{1}{3m})$. More recently, some new lower and upper bounds have been proposed in [3].

2. THE REDUCTION

Let **A** be an algorithm for $P||C_{max}$. Given a set of tasks

H, for every $i \leq m$, let $\mathbf{A}_i(H)$ be the set of tasks scheduled on M_i by the algorithm $\mathbf{A}$ and let $L_i(H) = \sum_{T_j \in \mathbf{A}_i(H)} l(T_j)$ denote the load of machine M_i. Let $C_{max}(H) = \max_i L_i(H)/s_i$ denote the makespan.

The reduction $\mathcal{T}(\mathbf{A})$:

Step 1: Assume that the machines have the same speed s_m, and apply the algorithm $\mathbf{A}$ on H. Let $L_i(H)$ be the sum of the lengths of the tasks scheduled on machine M_i, and let $C'_{max}(H)$ be the makespan of the obtained schedule.

Step 2: Add on each machine M_i, for $i = 1, \ldots, m$, a *dummy* task so that every machine finishes at time $C'_{max}(H)$. Let S_i be the set of tasks (the dummy task included) now scheduled on machine M_i.

Step 3: Do a random bijection between the set of tasks S_i (for $i \in \{1, \ldots, m\}$) and the machines $M_1, \ldots, M_m$. We consider now that each machine M_i has a speed s_i. (We thus obtain a new schedule of makespan $C_{max}(H) \leq r\, C'_{max}(H)$.)

Step 4: Then, on each machine, the tasks (including the dummy task) are scheduled in a random order (the execution of the dummy task corresponds to an idle time on its machine). Return the obtained schedule.

2.1 On-line version

It is possible to use the reduction $\mathcal{T}(\mathbf{A})$ in an on-line context in which the existence of a task is not known until its release date. Shmoys et al. [4] have shown that, for the related machine scheduling problem, it is possible to obtain an on-line 2ρ-approximation algorithm from any off-line ρ-approximation algorithm $\mathbf{A}$. The transformation works as follows:

Let S_0 be the set of tasks available at time 0, i.e. when the algorithm starts. The algorithm $\mathbf{A}$ is applied on the set S_0, and one obtains a schedule finishing at time F_0. Let S_1 be the set of tasks released in the time interval $(0, F_0]$. The algorithm $\mathbf{A}$ is applied on the set S_1 to obtain a schedule finishing at time F_1. In general, let S_{i+1} be the set of tasks released in $(F_{i-1}, F_i]$ with F_i the point in time when the tasks in S_i have been completed in the schedule returned by the algorithm $\mathbf{A}$. At time F_i the algorithm $\mathbf{A}$ is applied to schedule the tasks in S_{i+1}. If $S_{i+1} = \emptyset$, then all machines remain idle until the time t at which a new task is released; we then let $F_{i+1} = t$.

This technique can be applied to obtain an on-line truthful algorithm. Instead of considering the algorithm $\mathbf{A}$, we use the algorithm $\mathcal{T}(\mathcal{A})$. It is easy to see that the overall obtained on-line algorithm Online-$\mathcal{T}(\mathbf{A})$ is truthful for the weak model.

2.2 Increasing algorithms

Let H and H' be two sets of tasks $\{T_1, T_2, ..., T_n\}$ and $\{T'_1, T'_2, ..., T'_n\}$ respectively. We say that H' *dominates* H if and only if for every $1 \leq i \leq n$, one has $l(T'_i) \geq l(T_i)$ and we denote it by $H' \geq H$.

An algorithm A is said to be *increasing* if for every pair of sets of tasks H and H' such that $H' \geq H$, one has $C_{max}(H') \geq C_{max}(H)$.

3. MAIN RESULTS

1. Let $r = s_m/s_1$. The algorithm $\mathcal{T}(\mathbf{A})$ is a randomized truthful in expectation $(r \cdot \rho)$-approximation algorithm for $Q||C_{\max}$ in the *w-weak* model of execution provided that $\mathbf{A}$ is an increasing ρ-approximation algorithm for $P||C_{\max}$.
2. The algorithm Online-$\mathcal{T}(\mathbf{A})$ is an on-line truthful $(2r \cdot \rho)$-competitive algorithm for $Q||C_{\max}$ in the *w-weak* model of execution provided that $\mathbf{A}$ is an off-line increasing ρ-approximation algorithm for $P||C_{\max}$.
3. *LPT* (Largest Processing Time) is an increasing algorithm, therefore $\mathcal{T}(LPT)$ is a truthful $(4/3 - 1/3m)$-approximation algorithm for the *w-weak* model.
4. There is an increasing PTAS for $Pm||C_{max}$. Therefore, there is a randomized $r(1+\epsilon)$-approximation truthful algorithm for $Qm||C_{\max}$ in the *w-weak* model of execution.
5. Let $\mathbf{A}$ be an optimal algorithm. The reduction $\mathcal{T}(\mathbf{A})$ provides an equilibria-truthful algorithm in the *g-weak* model of execution.
6. If payments are allowed then our approach gives a $(1+\epsilon)$-approximation algorithm for the off-line case with related machines for the *w-weak* model of execution.

Remark. The main motivation for asking for truthful algorithms is that if the agents manipulate the scheduler by providing false information then even the most efficient algorithm may produce very bad results with respect to the global objective function. Our reduction $\mathcal{T}(\mathbf{A})$ directly links the individual function of the tasks to the global objective function. In the case of $P||C_{max}$, there is **no need to force the tasks to tell the truth**, because each of them will lie only if by doing so, the makespan decreases. Hence, if we relax the constraint that the algorithm must be truthful, then we can slightly modify our reduction by using an approximation algorithm $\mathbf{A}$ for $P||C_{max}$ which is not necessarily increasing. Then, the (modified) reduction $\mathcal{T}(\mathbf{A})$ is a truthful in expectation ρ-approximation algorithm provided that $\mathbf{A}$ is a ρ-approximation algorithm for $P||C_{max}$.

This opens a new interesting question in this area: *Is it always necessary to be truthful if our goal is just the optimization of the global objective function ?*

4. REFERENCES

[1] E. Angel, E. Bampis, and F. Pascual. Truthful algorithms for scheduling selfish tasks on parallel machines. *Theoretical Computer Science*, 369, 2006.

[2] V. Auletta, P. Penna, R. de Prisco, and P. Persiano. How to route and tax selfish unsplittable traffic. In *Proc. SPAA'2004*, pages 196–204, 2004.

[3] G. Christodoulou, L. Gourvès, and F. Pascual. Scheduling selfish tasks: About the performance of truthful algorithms. *Submitted.*

[4] D. Shmoys, J. Wein, and D.P. Williamson. Scheduling parallel machines on-line. *SIAM J. on Computing*, 24(6):1313 – 1331, 1995.