Algorithmic Mechanism Design

Noan Nisan, Amir Ronen
- 2nd Part -

Seminar Talk
by
Paul D. Dütting

Fakultät für Informatik
Universität Karlsruhe (TH)

May 3, 2007
Overview

Organization

Randomized Mechanisms
  Model of Randomization
  A Randomized Mechanism for Task Scheduling

Mechanisms with Verification
  Model of Verification
  The Compensation-and-Bonus Mechanism
  Polynomial-Time Mechanisms

Summary
Randomized Mechanisms

**Definition (Randomized Mechanism)**

A *randomized mechanism* is a probability distribution over a family \( \{ m_r | r \in I \} \) of mechanisms, all sharing the same sets of strategies and possible outputs. The outcome of such a mechanism is a probability distribution over outputs and payments; the problem specification must specify what output distributions are required. For the case of optimization problems, the objective function on such a distribution is taken to be the expectation \( g(a, t) = \mathbb{E}_{r \in I}(g(o_{m_r}(a), t)) \).
Dominant Strategies and Truthful Mechanisms

Definition (Universally Dominant Strategy)

A strategy $a^i$ is called universally dominant for agent $i$ if it is a dominant strategy for every mechanism in the support of the randomized mechanism. A randomized mechanism is called universally truthful if truth-telling is a dominant strategy, and strongly truthful if it is the only one.
Task Scheduling Problem Reviewed

Let’s recall the following key results:

**Theorem (Min Work Mechanism)**

*Min Work is a strongly truthful n-approximation mechanism for the task scheduling problem.*

**Theorem (Lower Bound for Deterministic Mechanisms)**

*There does not exist a mechanism that implements a c-approximation for the task scheduling problem for any c < 2.*

⇒ for 2 agents:
Min Work is a 2-approximation and no mechanism can do better
A Simple Example

Consider two kind of tasks $j = 1, 2$ and two agents $i = 1, 2$ such that $t_1^1 = 1$, $t_1^2 = 1 + \epsilon$, $t_2^1 = 10 + \epsilon$, $t_2^2 = 10$. If there are two tasks of each kind, the allocations of Min Work and Opt look like this:
Evaluating the Simple Example

Let’s look at the makespans:

\[ t_{\text{MinWork}} = 20 \quad t_{\text{Opt}} = 11 + \epsilon \implies \frac{t_{\text{MinWork}}}{t_{\text{Opt}}} = \frac{20}{11 + \epsilon} \approx 1.81 \text{ for small enough } \epsilon \]

Gets worse as the number of tasks or the size discrepancy grows!

Question

*Is it possible to achieve a similar behavior as Opt with a randomized mechanism? Should we allow some tasks to be allocated to the less efficient agent (as long as it is not much less efficient)?*
A Randomized Mechanism for Task Scheduling

Biased Min Work Mechanism

The Biased Min Work Mechanism (for 2 agents)

**Parameters:** A real number \( \beta \geq 1 \) and a bit vector \( s \in \{1, 2\} \)

**Input:** The reported type vectors \( t = (t_1, t_2) \)

**Output:** An allocation \( x = (x^1, x^2) \) & a payment \( p = (p^1, p^2) \)

**Mechanism:**

\[
\begin{align*}
x^1 & \leftarrow \emptyset; \\
x^2 & \leftarrow \emptyset, \\
p^1 & \leftarrow 0; \\
p^2 & \leftarrow 0 \\
\text{For each task } j = 1..k \text{ do:} & \\
& \text{Let } i = s_j \text{ and } i' = 3 - i \\
& \text{If } t^i_j \leq \beta \ t^{i'}_j \\
& \quad \text{Then } x^i \leftarrow x^i \cup \{j\}; \\
& \quad \quad p^i \leftarrow p^i + \beta \ t^{i'}_j \\
& \text{Else } x^{i''} \leftarrow x^{i''} \cup \{j\}; \\
& \quad \quad p^{i''} \leftarrow p^{i''} + \beta \ t^i_j
\end{align*}
\]
The Randomly Biased Min Work Mechanism

Definition (Randomly Biased Min Work Mechanism)
The Randomly Biased Min Work Mechanism is the distribution on biased min work mechanisms given by $\beta = 4/3$, and a uniform distribution of $s \in \{1, 2\}^k$.

Let's prove the following main result:

Theorem (Randomly Biased Min Work Mechanism)
The Randomly Biased Min Work Mechanism is a (polynomial time computable) strongly truthful implementation of a 7/4—approximation for task scheduling with two agents.
Proving the Theorem - Step 1

We will prove the preceding theorem in two steps.

**Step 1**

*The Randomly Biased Min Work Mechanism is strongly truthful.*

**Proof**

*See chalkboard!*
Proving the Theorem - Step 2

Step 2

The allocation obtained by the Randomly Biased Min Work Mechanism is a $7/4$— approximation for the task scheduling problem.

First, we will distillate the hardest case...
Claim

It is enough to consider the following case:

1. For each k-task the efficiency discrepancy between the agents is arbitrarily close to $\beta$.
2. If opt allocates an l-task $j$ to agent $i$, then $t_{j}^{3-i}/t_{j}^{i} = \beta$.
3. Under opt both agents have the same finishing time.
4. One of the agents is more efficient than the other on all k-tasks.
5. There are at most four tasks, where at most one k-task and at most one l-task is allocated by opt to each agent.
Proving the Theorem - The Hardest Case

Due to the preceding considerations we can concentrate on the following case:

<table>
<thead>
<tr>
<th></th>
<th>$t^1_j$</th>
<th>$t^2_j$</th>
<th>opt-alloc</th>
<th>bmw-alloc</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1$</td>
<td>$a$</td>
<td>$\beta a$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$k_2$</td>
<td>$b$</td>
<td>$\beta b$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$l_1$</td>
<td>$c$</td>
<td>$\beta c$</td>
<td>1</td>
<td>rnd</td>
</tr>
<tr>
<td>$l_2$</td>
<td>$\beta d$</td>
<td>$d$</td>
<td>2</td>
<td>rnd</td>
</tr>
</tbody>
</table>

Proof

See chalkboard!
A Randomized Mechanism for Task Scheduling

Intermediate Summary

So far...

- we introduced randomized mechanisms and transferred the notions dominant strategy and (strongly) truthful mechanism to this kind of mechanisms.
- we devised a randomized mechanism for the task scheduling problem with two agents that beats the lower bound for deterministic mechanisms.
Mechanisms with Verification - The Basic Idea

Idea

*Up to now the sole information the mechanisms are based upon are the agents’ declarations. In many settings though, one could take advantage of various extra information available at varying points in time.*

We will focus on:

**Mechanisms with 2 Stages**

*Declaration Phase: agents talk $\rightarrow$ decision (eg. allocation)*

*Execution Phase: agents execute the agreed output $\rightarrow$ payment*
Mechanism with Verification

Definition (Mechanism with Verification 1/2)

- An agent’s strategy is composed of two separate parts: a declaration $d^i$ and an execution $e^i$.
- Each declaration $d^i$ is chosen by the agent, based on its type $t^i$, in an unrestricted manner.
- The decision $k$ of the mechanism must be a function of just the declarations $d^1, .., d^n$.
- The agent’s execution $e^i$ may depend on $t^i$ as well as on $k$. The problem specification specifies, for each $t^i$, the possible $e()$’s an agent of type $t^i$ may choose.
Mechanism with Verification

Definition (Mechanism with Verification 2/2)

- The output of the mechanism is the result of the decision $k$ and the agents' executions $e^1(k), .., e^n(k)$. The output function $o(k, e)$ is a part of the problem specification.
- The output $o$, determines both the objective function $g(o, t)$ and the agents' valuations' $v^i(t^i, o)$.
- The payment $p^i$ that the mechanism provides depends on both, the declarations $d^1, .., d^n$ and the executions $e^1(k), .., e^n(k)$. 
Truthful Mechanism with Verification

Definition (Truthful Mechanism with Verification)
A mechanism with verification is called truthful if

1. The agents’ declarations are simply to report their types.
2. For each agent $i$ of type $t^i$, there is a dominant strategy of the form $a^i = (t^i, e^i())$.

We say that the mechanism is strongly truthful if it is the only dominant strategy.
Definition (Task Scheduling with Verification)

The problem is the same as before, except that the mechanism knows the times \( \tilde{t} = \tilde{t}_1, \ldots, \tilde{t}_k \) the tasks were actually performed and may base its payment \( p(t, \tilde{t}) = p^1(t, \tilde{t}), \ldots, p^n(t, \tilde{t}) \) on this extra information.

- Mechanism: \((x(t), p)\) where \( x(t) \) is the allocation function
- Feasible output: \((x, \tilde{t})\) where \( x = x^1, \ldots, x^n \) is the allocation
- Objective function: \( g(x, \tilde{t}) = \max_i \sum_{j \in x^i} \tilde{t}_j \) (make-span)
- Agent \( i \)'s valuation: \( v^i(x, \tilde{t}) = -\sum_{j \in x^i} \tilde{t}_j \)
- Agent \( i \)'s utility: \( u^i(t, \tilde{t}) = p^i(t, \tilde{t}) + v^i(x, \tilde{t}) \)
The Compensation-and-Bonus Mechanism

The crucial aspect is to choose the payments adequately:

**Definition (Compensation-and-Bonus Mechanism)**

The **Compensation-and-Bonus Mechanism** is defined by an optimal allocation function \( x(t) \) and the payment functions

\[
p^i(t, \tilde{t}) = c^i(t, \tilde{t}) + b^i(t, \tilde{t})
\]

with

- **Compensation:**
  \[
c^i(t, \tilde{t}) = \sum_{j \in x_i} \tilde{t}_j
\]

- **Corrected time vector:**
  \[
corr^i(x, t, \tilde{t}) = \begin{cases} 
  t^l_j & \text{if } j \in x^l \text{ and } l \neq i \\
  \tilde{t}_j & \text{if } j \in x^i
\end{cases}
\]

- **Bonus:**
  \[
b^i(t, \tilde{t}) = -g(x(t), corr^i(x, t, \tilde{t}))
\]
Example 1: Truth-telling and Eager Agents

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent 1</td>
<td>$t^1_j$</td>
<td>10</td>
<td>30</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}^1_j$</td>
<td>10</td>
<td>30</td>
<td>45</td>
</tr>
<tr>
<td>Agent 2</td>
<td>$t^2_j$</td>
<td>100</td>
<td>60</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}^2_j$</td>
<td>100</td>
<td>60</td>
<td>100</td>
</tr>
</tbody>
</table>

Compensation: $c^i(t, \tilde{t}) = \sum_{j \in x^i} \tilde{t}_j \Rightarrow c^1(\ldots) = 55$ and $c^2(\ldots) = 60$

Corr. time vec.: $corr^i(x, t, \tilde{t}) = \begin{cases} t^l_j & \text{if } j \in x^l \text{ and } l \neq i \\ \tilde{t}_j & \text{if } j \in x^i \end{cases} \Rightarrow corr^{1,2}(\ldots) = \tilde{t}$

Bonus: $b^i(t, \tilde{t}) = -g(x(t), corr^i(x, t, \tilde{t})) \Rightarrow b^{1,2}(\ldots) = -60$

Payment: $p^i(t, \tilde{t}) = c^i(t, \tilde{t}) + b^i(t, \tilde{t}) \Rightarrow p^1(\ldots) = -5 \& p^2(\ldots) = 0$

Utilities: $u^i(t, \tilde{t}) = p^i(t, \tilde{t}) + v^i(x, \tilde{t}) \Rightarrow u^1(\ldots) = u^2(\ldots) = -60$
Example 2: Lying but Eager Agents

<table>
<thead>
<tr>
<th>Agent</th>
<th>$t_j$</th>
<th>$\tilde{t}_j$</th>
<th>$\tilde{t}'_j$</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent 1</td>
<td>10</td>
<td>30</td>
<td>200</td>
<td>opt.alloc.</td>
</tr>
<tr>
<td>Agent 2</td>
<td>100</td>
<td>60</td>
<td>100</td>
<td>opt.alloc.</td>
</tr>
</tbody>
</table>

Compensation: $c^1(.) = 40$ and $c^2(.) = 100$

Corrected time vector: $corr^{1,2}(.). = \tilde{t} = (10, 30, 100)$

Bonus: $b^{1,2}(.). = -100$

Payment: $p^1(.) = -60$ & $p^2(.) = 0$

Utilities: $u^1(.) = u^2(.) = -100$
**Example 3: Truth-telling but Lazy Agents**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent 1</td>
<td>$t_j^1$</td>
<td>10</td>
<td>30</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_j^1$</td>
<td>20</td>
<td>40</td>
<td>80</td>
</tr>
<tr>
<td>Agent 2</td>
<td>$t_j^2$</td>
<td>100</td>
<td>60</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>$\tilde{t}_j^2$</td>
<td>100</td>
<td>60</td>
<td>100</td>
</tr>
</tbody>
</table>

Compensation: $c^1(\ldots) = 40$ and $c^2(\ldots) = 100$

Corr. time vec.: $corr^1(\ldots) = (20, 60, 80)$ & $corr^2(\ldots) = (10, 60, 45)$

Bonus: $b^1(\ldots) = -100$ and $b^2(\ldots) = -60$

Payment: $p^1(\ldots) = -60$ and $p^2(\ldots) = 0$

Utilities: $u^1(\ldots) = -100$ and $u^2(\ldots) = -60$
Truthfulness of the Compensation-and-Bonus Mechanism

Theorem (Compensation-and-Bonus Mechanism)

The Compensation-and-Bonus Mechanism is a strongly truthful implementation of the task scheduling problem.

Proof.

See chalkboard!
The Generalized Compensation-and-Bonus Mechanism

The preceding findings can be generalized as follows:

Definition (Generalized Compensation-and-Bonus Mechanism)  
The Generalized Compensation-and-Bonus Mechanism is defined by an optimal allocation function $x(t)$ and the payment functions $p^i(t, \tilde{t}) = c^i(t, \tilde{t}) + b^i(t, \tilde{t})$ with

- Generalized Compensation: $c^i(t, \tilde{t}) \leq \sum_{j \in x_i} \tilde{t}_j$
- Generalized Bonus: $b^i(t, \tilde{t}) = m^i(t^{-i}, -g(x(t), corr^i(x, t, \tilde{t})))$ where $m^i(t^{-i}, w)$ is any positive real-valued function, that is monotonically increasing in $w$
Further Findings

**Theorem (Generalized Compensation-and-Bonus Mechanism)**

*The Generalized Compensation-and-Bonus Mechanism is a strongly truthful implementation of the task scheduling problem.*

**Annotation**

*The Generalized Compensation-and-Bonus Mechanism can be used to satisfy*

- **Participation Constraints:** If an agent is truth-telling, its utility is non-negative.
- **Budget Limits:** The total payments may not exceed a certain maximum amount specified in advance.
The Compensation-and-Bonus Mechanism

Intermediate Summary

So far...

- *We introduced an extended model*, formalizing situations where the mechanism has to make a decision based on the agents’ declarations but delays their payments until it knows their actual behavior.

- *We devised two optimal mechanisms* for the adapted task scheduling problem. The key idea was to calculate an optimal allocation of tasks based on the agents’ declarations and to determine their payments based on the actual execution times. *This allows cheating agents to be punished.*
Motivation

The rest of this talk is concerned with the following problem:

Problem

Computing an optimal allocation is known to be NP-hard.

Idea

One could replace the optimal allocation algorithm in the Compensation-and-Bonus Mechanism with a known polynomial-time approximation algorithm.
Our First Idea Fails

Unfortunately, this does not work as the following theorem shows:

**Theorem (Approximation Lacks Truthfulness)**

Any Compensation-and-Bonus Mechanism that is based on a non-optimal allocation algorithm is not truthful.

**Proof.**

See chalkboard!
Let’s look at another variant of the task scheduling problem!

**Definition (Bounded Scheduling Problem)**

The problem is the same as in task scheduling with verification, except that the number of agents $n$ is fixed to a constant and there exist fixed constants $a, b > 0$ such that $a \leq t^i_j \leq b$ for all $i$ and $j$.

**Aim**

*Design a $(1 + \epsilon)-approximation mechanism for this restricted variant of the problem.*
The Rounding Algorithm of Horowitz and Sahni

Idea

*Use the following approximation algorithm for the allocation of tasks (see Horowitz and Sahni [1] for details).*

The Rounding Algorithm of Horowitz and Sahni

- *Round up the entries* $t_{ij}$ *to integer multiples of* $\delta$ *(a parameter chosen as a function of* $a$ *and* $\epsilon$ *).*
- *Solve this rounded problem exactly in polynomial time using dynamic programming.*
Polynomial-Time Mechanisms

The Payment Function of Nisan and Ronen

Idea

*Use the following payment function.*

The Payment Function of Nisan and Ronen

Let $\hat{t}$ denote the vector where all entries of $t$ are rounded up to an integer multiple of $\delta$. Let $g(.)$ be the make-span function and denote by $\hat{g}(x, t)$ the make-span $g(x, \hat{t})$. The payment function is given by $p^i(t, \hat{t}) = c^i(t, \hat{t}) + b^i(t, \hat{t})$, where

- $c^i(t, \hat{t}) = \sum_{j \in i(t)} \hat{t}_j$ is the Compensation and
- $b^i(t, \hat{t}) = -\hat{g}(x(t), corr^i(x(t), t, \hat{t}))$ is the Bonus.
The Rounding Mechanism

Theorem
The Rounding Mechanism uses the Rounding Algorithm of Horowitz and Sahni for the allocation of tasks and the Payment Function of Nisan and Ronen. For every fixed $\epsilon > 0$ it is a polynomial time mechanism with verification that truthfully implements a $(1 + \epsilon)$-approximation for the Bounded Task Scheduling problem.

Proof.
See chalkboard!
Intermediate Summary

So far...

- we pointed out that computing the optimal allocation of tasks is NP-hard. As a consequence the optimal Compensation-and-Bonus Mechanism is computationally intractable. Simply replacing the optimal allocation algorithm by an approximation algorithm does not work in general.

- we introduced a restricted variant of the task scheduling with verification problem. For this problem we devised a polynomial time $(1 + \epsilon)$—approximation mechanism.
Summary (of the 2nd Part)

- We designed a randomized mechanism for the task scheduling problem that beats the lower bound for deterministic mechanisms.
- We proposed an extended model, mechanisms with verification, and presented two (computationally intractable) optimal mechanisms.
- For a restricted variant of the task scheduling problem we devised a \((1 + \epsilon)-\)approximation mechanism.
That’s it. Thank you!
Are there any questions?
Literature

