Algorithmic Mechanism Design

N. Nisan, A. Ronen

Adam Bains

October 22, 2008

- Designers of communication protocols typically assume that agents are trustworthy...

- Designers of communication protocols typically assume that agents are trustworthy...
- ... but what about those that aren't?
  - Faulty or compromised computers
  - Malicious agents
- Authors of the paper are primarily concerned with examining mechanism design from a CS standpoint

- Some elements have already been covered in class:
  - Basic mechanism design
  - Time-complexity considerations
- How is this different from what we've previously seen?
  - Different mechanism design models
  - Incentive-compatibility results for approximation algorithms

A quick review...

- **Recall:** Mechanisms attempt to coordinate multiple rational agents in order to solve a problem
  - Goal represented by a social choice function $f : \Theta_1 \times \ldots \times \Theta_n \rightarrow O$
  - Mechanism defined as $M = (S_1, \ldots, S_n, g(s))$, where $g : S_1 \times \ldots \times S_n \rightarrow O$

We're going to be focusing on VCG mechanisms, which means that the mechanisms we consider will have:

- Direct-revelation
  - i.e. $\forall i$, $S_i = \Theta_i$
- Quasi-linear preferences
  - $u_i = v_i(o, \theta_i) + p_i$, where v_i is agent valuation, p_i is mechanism's payment function

But how do we apply VCG mechanisms to A&C problems?

Basic Example: Min. Weighted Path

Problem: Given target nodes $x, y \in G$, we need to find the minimum weight path from $x \rightarrow y$, where each edge e is an agent whose edge cost is private.

- Edge cost for e_i is $\theta_i \geq 0$
- $v_i(o, \theta_i)$ is $-\theta_i$ if used (0 otherwise)

If agents lie about their edge weights, we can't find the optimal path; need to create a payment function that promotes truth-telling:

$$p_i = d_{G|i=\infty} - d_{G|i=0}$$

Where $d_{G|i=\infty}$ is the weight of min. weight path that doesn't use i, and $d_{G|i=0}$ is the weight of the path with $\theta_i = 0$.

Definition: A direct-revelation mechanism is a VCG mechanism if:

1. The outcome function maximizes overall agent valuation
2. The payment function for agent i is a combination of the sum of other agents' valuations, plus $h_i(\theta_{-i})$, which is an arbitrary function of other agents' types.

With this in mind, the mechanism that we are discussing is clearly a VCG mechanism:

- $d_{G|i=\infty}$ is equivalent to $h_i(\theta_{-i})$
- $d_{G|i=0}$ is the same as $\sum_{j \neq i} v_j(\theta_j, o(t))$.

This was a fairly simple example, but it has shown us:

- how to apply VCG mechanisms to standard A&C problems.
- a template for possible future VCG-based solutions, e.g. minimum spanning tree.

Definition: Task Allocation Problem

- **Goal**: Assign k tasks to n agents such that the completion time is minimized
  - Set of feasible outputs is the set of all possible task partitions
  - Task partition $x = x_1, \ldots, x_n$ is an n-tuple of (possibly empty) sets x_i, where x_i is the set of tasks allocated to i
  - Objective function is the completion time of the final task $(\max_i \sum_{j \in x_i} \theta_i^j)$

Agent properties:

- Agent type θ_i determines θ_i^j, the amount of time an agent of type i requires to complete task j
- Agent valuation is the negation of the sum of the time to complete all tasks assigned to it
    - Formally: $v_i(x, \theta_i) = -\sum_{j \in x_i} \theta_i^j$

The naïve approach:

To start, consider a simple approximation of the task allocation problem: the minimum work mechanism

- **Idea**: attempt to minimize the total amount of work done
  - Allocate each task j to the agent with min θ^j
- Not a close approximation – consider cases where all tasks are allocated to one agent
- Can be used to develop an upper bound on the task allocation problem

The idea behind the mechanism is fairly simple:

- Optimal allocation is simple: just choose the agent with the smallest θ^j for each task.
- Payment is simply the second-best time for each task assigned to that agent.

Given a mechanism satisfying this problem that is also in the family of VCG mechanisms:

- Outcome is at most the sum of the minimum θ^j values for $j = 1 \ldots n$
  - More formally, $g(x(t), t) \leq \sum_{j=1}^{k} \min_i \theta_i^j$
  - Assumes tasks are performed sequentially
- Optimal value is at least $1/n$ times this value

Lower Bound: Task Scheduling Problem

Theorem

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

Proof: Consider scenario with 2 agents, $k \geq 3$ tasks, $|x_1(\theta)| \leq |x_2(\theta)|$, and $0 < \epsilon < 1$.

- Let $x_1(\hat{\theta})$ be $x_1(\theta)$ with $\theta_1^j = \epsilon$ for all $j \in x_1(\theta)$
- Let $x_2(\hat{\theta})$ be $x_2(\theta)$ with $\theta_1^j = 1 + \epsilon$ for all $j \in x_2(\theta)$
- Then $x(\hat{\theta}) = x(\theta)$

- As type of agent 2 is unchanged, prices offered remain the same, so $x_1(\hat{\theta})$ is an identical task allocation.
- Since we are dealing with a two agent scenario, $x_2(\hat{\theta}) = x_2(\theta)$.
- Finally, assuming $|x_2(\theta)|$ is even, the optimal allocation on the adjusted task vectors is at most $\frac{1}{2}|x_2| + k\epsilon$.
- The odd case follows a similar proof. $\square$

Paper also presents additional results for the task scheduling problem:

- Tight upper bound for additive mechanisms
- Tight upper bound for local mechanisms
- Approximation mechanism that circumvents the lower bound for task scheduling

Are there any problems with the existing model?

- Doesn't make any assumptions about agent strategies – allows agents to report any $\theta \in \Theta$, regardless of actual type
- Models the communication phase – agents communicate, mechanism assigns tasks...

Are there any problems with the existing model?

- Doesn't make any assumptions about agent strategies – allows agents to report any $\theta \in \Theta$, regardless of actual type
- Models the communication phase – agents communicate, mechanism assigns tasks...
- ... but what about the actual execution of the tasks?

Are there any problems with the existing model?

- Doesn't make any assumptions about agent strategies – allows agents to report any $\theta \in \Theta$, regardless of actual type
- Models the communication phase – agents communicate, mechanism assigns tasks...
- ... but what about the actual execution of the tasks?
- Authors propose a new model, mechanisms with verification, that takes this into account.

Why take execution into account?

- Allows us to observe difference between reported type and actual type
- Can withhold payment until after execution, basing it on actual performance instead of declared performance
  - Seems to model real-world concerns more closely

Definitions

- Agent strategies now have two components: declaration (d_i) and execution ($e_i()$)
  - e_i depends on both θ_i and outcome $x(d)$
  - Both d_i and $e_i(x)$ used to determine payment
  - Still representing minimum time to completion for task j with θ_i^j; actual time represented by $\tilde{\theta}^j$
- Output is now $o(x, e)$ – depends on task allocation and execution times

The payment function

The payment function is broken into two components, resulting in a *Compensation-and-Bonus* mechanism.

- Payment function is the sum of the compensation function and the bonus function
- *Compensation function* is the sum of all actual execution times (i.e. $\sum_{j \in x_i(\theta)} \tilde{\theta}^j$))

- *Bonus function* is the negation of the maximum completion time in i's corrected time vector (i.e. $-g(x(\theta), corr_i(x(\theta), \theta, \tilde{\theta}))$
    - Corrected time vector for i ($corr_i(x(\theta), \theta, \tilde{\theta})$) is the set of all declared execution times, with i's declared times (θ_i^j) replaced with actual times ($\tilde{\theta}_i^j$)
    - Bonus function depends on i's execution times and declared times for all other agents

Proof of Truthfulness

Essentially:

- The payment described above is maximized when the agent executes its assignments in minimal time – increasing execution time would only decrease the bonus value.

Proof of Truthfulness

Essentially:

- The payment described above is maximized when the agent executes its assignments in minimal time – increasing execution time would only decrease the bonus value.
- As we're working with an optimal allocation algorithm, we already know that the final completion time $(g(x(\theta), corr_i(x(\theta), \theta, \tilde{\theta}))$ is minimized.

Proof of Truthfulness

Essentially:

- The payment described above is maximized when the agent executes its assignments in minimal time – increasing execution time would only decrease the bonus value.
- As we're working with an optimal allocation algorithm, we already know that the final completion time $(g(x(\theta), corr_i(x(\theta), \theta, \tilde{\theta}))$ is minimized.
- Since a minimization of this value is equal to a maximization of the bonus, we know that reporting any other type will only decrease agent i's bonus, or at best leave it unchanged.

Proof of Truthfulness

Essentially:

- The payment described above is maximized when the agent executes its assignments in minimal time – increasing execution time would only decrease the bonus value.
- As we're working with an optimal allocation algorithm, we already know that the final completion time $(g(x(\theta), corr_i(x(\theta), \theta, \tilde{\theta}))$ is minimized.
- Since a minimization of this value is equal to a maximization of the bonus, we know that reporting any other type will only decrease agent i's bonus, or at best leave it unchanged.
- Therefore, truth-telling is the only dominant strategy.

Problem: Intractability

- Previous mechanism approaches mentioned rely on *optimal allocation algorithm*, but this isn't computationally feasible
- How do we get around this?

Poly-Time Approximation Algorithms

- **Question:** What happens when we replace optimal allocation with a poly-time algorithm?

Theorem

Let $x()$ be a non-optimal approximation algorithm for task scheduling. Let $m = (x, p)$ be the Compensation-and-Bonus mechanism based on $x()$. Then m is not truthful.

Poly-Time Approximation Algorithms

- **Question:** What happens when we replace optimal allocation with a poly-time algorithm?

Theorem

Let $x()$ be a non-optimal approximation algorithm for task scheduling. Let $m = (x, p)$ be the Compensation-and-Bonus mechanism based on $x()$. Then m is not truthful.

- ... but why?

Proof: *By contradiction*. Consider the case for Compensation-and-Bonus mechanisms. Assume that the approximate allocation mechanism is truthful:

- Let $x(\theta)$ represent the non-optimal approximation, and $opt(\theta)$ the optimal
- Let θ'_1 be a type for agent 1 such that time to completion (θ'^j_1) is the same as θ^j_i if task j was in the optimal task allocation, and an arbitrarily high value otherwise.
- Let $\theta' = \theta'_1, \theta_2, \ldots, \theta_n$

- We already know that $g(opt(\theta), \theta) < g(x(\theta), \theta)$, as $x()$ is a non-optimal approximation
- From the above definition of θ', we know that $g(x(\theta'), \theta) \geq g(x(\theta), \theta)$, as otherwise agent 1 would lie about its type, declaring θ'_1
- Apply this to all type vectors, and call the resulting set s

- We know that $g(x(s), \theta) \geq g(x(\theta), \theta)$ by the above argument.
- However, it is also clear that $g(x(s), \theta) = g(opt(s), \theta)$.
- As we already know that $g(x(s), \theta) \geq g(opt(s), \theta)$, $x(s)$ can't have the same allocation as $opt(s)$ – there must be some task j that is allocated to a different agent in $x()$.
- This contradicts the algorithm's approximation, as the completion time for task j will be ∞ for $x()$'s allocation. Thus, we have demonstrated via contradiction that it cannot be truthful.

- Proof presented in this paper deals specifically with *Compensation-and-Bonus* mechanisms
- [NR00] examines the issue in a more general context

Recap

In this paper, the authors presented:

- Upper- and lower-bounds on approximation for the task scheduling mechanism
- An extended mechanism design model (*Compensation-and-Bonus*) restricting agent actions to reflect their type.
- Proof that Compensation-and-Bonus-based approximation mechanisms are not incentive- compatible

Takeaway Messages

- Standard mechanism design is fine in theory, but computationally intractable in practice
- Optimal approximation mechanisms are not always possible

Future Work

The paper was very fundamental, so there were plenty of avenues for possible research:

- Different model extensions: examine other equilibrium types, game types, agent strategy types
- Further examine upper- and lower-bounds on examples presented, or examine applications to other problems
- Mechanism construction and implementation

Noam Nisan and Amir Ronen.
Computationally feasible vcg mechanisms.
In *EC '00: Proceedings of the 2nd ACM conference on Electronic commerce*, pages 242–252, New York, NY, USA, 2000. ACM.