An Antisocial Strategy for Scheduling Mechanisms

Nandan Garg   Daniel Grosu   Vipin Chaudhary

Dept. of Computer Science, Wayne State University
5143 Cass Avenue, Detroit, MI 48202
nandang@wayne.edu, dgrosu@cs.wayne.edu, vipin@wayne.edu

Abstract

Previous work on task scheduling mechanisms assumed that the agent's goal is to maximize its own profit without considering the effect of its strategy on the other agents' profit. This is not always the case, an agent may want to cause loses to the other agents besides maximizing its profit. Such an agent is said to be an antisocial agent. An antisocial agent will try to gain as much profit as possible relative to the other agents. In this paper we consider a mechanism for task scheduling on related machines in which each machine is associated with an agent. We develop an antisocial strategy which can be used by an antisocial agent to inflict losses to the other participating agents. We analyze the effect of different degrees of agent's antisociality on the losses inflicted to the other agents.

1. Introduction

The current distributed computing systems such as computational grids are composed of geographically distributed resources (computers, storage etc.) owned by self interested agents or organizations. These agents may manipulate the load allocation protocol in their own benefit and their selfish behavior may lead to severe performance degradation and poor efficiency. Solving such problems involving selfish agents is the object of *mechanism design theory* (also called *implementation theory*) [14]. This theory helps design protocols in which the agents are always forced to tell the truth and follow the rules. Such mechanisms are called *truthful* or *strategyproof*. In a mechanism each participant has a privately known function called *valuation* which quantifies the agent's benefit or loss. Payments are designed and used to motivate the participants to report their true valuations. The goal of each participant is to maximize the difference of its valuation and payment. An agent makes profit if the payment received by the agent is greater than its true valuation and it suffers losses if the payment is less than its true valuation. As an example consider several resource providers that offer computer services. We assume that each resource is characterized by its job processing rate. An allocation mechanism is strategyproof if a resource owner maximizes her utility only by reporting the true resource processing rate to the mechanism. The optimal utility is independent of the values reported by the other participating resource owners.

Previous work on task scheduling mechanisms assumed that the agent's goal is to maximize its own profit without considering the effect of its strategy on the other agents' profit [9, 13]. This is not always the case, an agent may want to inflict loses to other agents rather than maximize its own profit. Such an agent is said to be an antisocial agent. An antisocial agent will try to gain as much profit as possible relative to the other agents. An antisocial agent needs to determine a bidding strategy that will allow it to inflict losses to the other participating agents while causing minimum losses to itself. Also given that the true value of an agent is a private value, the antisocial agent needs a way to infer the true values of the other agents such that it can modify its bid accordingly.

In this paper we consider a mechanism for task scheduling on related machines and develop an antisocial strategy which can be used by an antisocial agent (associated with one machine) to inflict losses to the other participating agents. The strategy is proposed for a mechanism where the tasks with different sizes are to be allocated by the mechanism. We analyze the effect of different agent's parameters on the losses inflicted to the other agents.

Related work. Recently many researchers have used mechanism design theory to solve problems in areas like resource allocation and task scheduling [12, 17, 18], congestion control and routing [5, 11]. Nisan and Ronen [13] studied different mechanisms for shortest path and task scheduling. They proposed mechanisms to solve the shortest path problem and the problem of task scheduling on unrelated machines based on the popular VCG (Vickrey-Clarke-Groves) mechanism [4, 10, 16]. VCG mechanisms can be applied only to problems where the objective functions are simply

the sum of agent's valuations and the set of outputs is finite. A general framework for designing truthful mechanisms for optimization problems where the agents' private data is one real valued parameter was proposed by Archer and Tardos [1]. Their framework can be applied to design mechanisms for optimization problems with general objective functions and restricted form of valuations. They also studied the frugality of shortest path mechanisms in [2]. A truthful mechanism that gives the overall optimal solution for the static load balancing problem in distributed systems is proposed in [9]. Feigenbaum *et al.* [6] studied the computational aspects of mechanisms for cost sharing in multicast transmissions. A mechanism for low cost routing in networks is proposed in [5]. The results and the challenges of designing distributed mechanisms are surveyed in [7]. Goldberg and Hartline [8] studied the problem of designing mechanisms such that no coalition of agents can increase the combined utility of the coalition by engaging in a collusive strategy. The closest work to the present study is the work of Brandt and Weiss [3] in which the authors studied the behavior of antisocial agents in Vickrey auctions. They considered repeated Vickrey auctions in which the same item is auctioned in each round. They derived an antisocial strategy and introduced some notations to formalize the study of antisocial agents. Our paper considers a different scenario where the tasks (items) are different but related in terms of their execution times.

Our contributions. We consider a mechanism for scheduling on related machines and develop an antisocial strategy for the participating agents. We characterize the antisociality of an agent and analyze the effect of different degrees of antisociality on the loss an agent can inflict to the other agents. We simulate the strategy and verify the correctness of the strategy. By means of simulation we also study the effect of changing different parameters on the amount of losses an antisocial agent can inflict on the other participating agents.

Organization. In Section 2 we present the formal model of the task scheduling problem and the scheduling mechanism. In Section 3 we present a formal characterization of the antisocial behavior and then present the antisocial strategy for the scheduling mechanism. In Section 4 we present and discuss experimental results. Section 5 concludes the paper.

2. The Scheduling Problem and Mechanisms

2.1 The Scheduling Problem

We consider here the problem of scheduling $m \geq 1$ independent tasks $T_1, T_2, \ldots, T_m$ on $n \geq 1$ agents (machines)
$A_1, A_2, \ldots, A_n$. Each task T_j is characterized by its processing requirement of r_j units of time. Each agent A_i is characterized by its processing speed $s_i > 0$ and thus task T_j would take $t_j^i = r_j/s_i$ time to be processed by A_i. A schedule $\mathbf{S}$ is a partition of the set of tasks indices into disjoint sets S^i, $i = 1, \ldots, n$. Partition S^i contains the indices corresponding to all the tasks allocated to A_i. The goal is to obtain a schedule $\mathbf{S}$ minimizing a given objective function such as makespan, sum of completion times, etc. In the scheduling literature, this problem is known as *scheduling on related machines* [15].

2.2 Scheduling Mechanisms

In this section we first describe the scheduling mechanism design problem and then present two scheduling mechanisms.

Definition 2.1 (Mechanism design problem) The problem of designing a scheduling mechanism is characterized by:

(i) A finite set $\mathcal{S}$ of allowed outputs. The output is a schedule $\mathbf{S}(\mathbf{b}) = (S^1(\mathbf{b}), S^2(\mathbf{b}), \ldots, S^n(\mathbf{b}))$, $\mathbf{S}(\mathbf{b}) \in \mathcal{S}$, computed according to the agents' bids, $\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n)$. Here, $\mathbf{b}_i = (b_1^i, b_2^i, \ldots, b_m^i)$ is the vector of values (bids) reported by agent A_i to the mechanism.

(ii) Each agent A_i, $(i = 1, \ldots, n)$, has for each task T_j a privately known parameter t_j^i $(j = 1, \ldots, m)$ called the *true value* which represents the time required by agent A_i to execute task T_j. The preferences of agent A_i are given by a function called *valuation* $V_i(\mathbf{S}(\mathbf{b}), \mathbf{t}_i) = \sum_{j \in S^i(\mathbf{b})} t_j^i$ (i.e. the total time it takes to complete all tasks assigned to it). Here $\mathbf{t}_i = (t_1^i, t_2^i, \ldots, t_m^i)$.

(iii) Each agent goal is to maximize its *utility*. The utility of agent A_i is $U_i(\mathbf{b}, \mathbf{t}) = P_i(\mathbf{b}, \mathbf{t}) - V_i(\mathbf{S}(\mathbf{b}), \mathbf{t}_i)$, where P_i is the payment handed by the mechanism to agent A_i.

(iv) The goal of the mechanism is to select a schedule $\mathbf{S}$ that minimizes the make-span $C_{max} = \max_i \sum_{j \in S^i(\mathbf{b})} t_j^i$.

An agent A_i may report a value (bid) b_j^i for a task T_j different from its true value t_j^i. The true value characterizes the actual processing capacity of agent A_i. The goal of a truthful scheduling mechanism is to give incentives to agents such that it is beneficial for them to report their true values. Now we give a formal description of a mechanism and define the concept of truthful mechanism.

Definition 2.2 (Mechanism) A *mechanism* is a pair of functions:

(i) The *allocation function*: $\mathbf{S}(\mathbf{b}) = (S^1(\mathbf{b}), S^2(\mathbf{b}), \ldots, S^n(\mathbf{b}))$. This function has as input the vector of agents' bids $\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n)$ and returns an output $\mathbf{S} \in \mathcal{S}$.

(ii) The *payment function*: $P(\mathbf{b}, \mathbf{t}) = (P_1(\mathbf{b}, \mathbf{t}), P_2(\mathbf{b}, \mathbf{t}), \ldots, P_n(\mathbf{b}, \mathbf{t}))$, where $P_i(\mathbf{b}, \mathbf{t})$ is the payment handed by the mechanism to agent A_i.

Definition 2.3 (Truthful mechanism) A mechanism is called *truthful* or *strategyproof* if for every agent A_i of type $\mathbf{t}_i$ and for every bids $\mathbf{b}_{-i} = (\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}, \mathbf{b}_{i+1}, \ldots, \mathbf{b}_n)$ of the other agents, the agent's utility is maximized when it declares its real value $\mathbf{t}_i$ (i.e. truth-telling is a dominant strategy).

Nisan and Ronen [13] designed a truthful mechanism for a more general problem called scheduling on unrelated machines. In this case the speed of machines depends on the task i.e. for a task T_j and agent A_i, the processing speed is s_{ij}. If $s_{ij} = s_i$ for all i and j then the problem reduces to the problem of scheduling on related machines [15]. The authors provided an approximation mechanism called Min-Work, minimizing the total amount of work. MinWork is a truthful mechanism. In the following we present the Min-Work mechanism.

Definition 2.4 (MinWork Mechanism) [13] The mechanism that gives an approximate solution to the scheduling problem is defined by the following two functions:

(i) The *allocation function*: each task is allocated to the agent who is able to execute the task in a minimum amount of time (Allocation is random when there are more than one agent with minimum type).

(ii) The *payment function* for agent A_i given by:

$$P_i(\mathbf{b}, \mathbf{t}) = \sum_{j \in S^i(\mathbf{b})} \min_{i' \neq i} t_j^{i'} \qquad (1)$$

The MinWork mechanism can be viewed as running separate Vickrey auctions simultaneously for each task. In this paper we consider a modification of MinWork mechanism that solves the problem of scheduling on related machines. This mechanism can be viewed as running a sequence of separate Vickrey auctions, one for each task. Thus the main difference from MinWork is that auctions are run in sequence and not simultaneously.

Definition 2.5 (Modified MinWork Mechanism) The mechanism that gives an approximate solution to the problem of task scheduling on related machines is defined as follows:

For each task T_j ($j = 1, 2, \ldots, m$):

(i) The *allocation function*: Task T_j is allocated to the agent who is able to execute it in a minimum amount of time.

(ii) The *payment function* for agent A_i given by:

$$P_i(\mathbf{b}, \mathbf{t}) = \min_{i' \neq i} t_j^{i'}, \qquad j \in S^i(\mathbf{b}) \qquad (2)$$

In the following we present the protocol that implements the Modified MinWork mechanism (MMW).

Protocol MMW:

For each task T_j, $j = 1, \ldots, m$:

1. Agent A_i, $i = 1, \ldots, n$ submits bid b_j^i to the mechanism.

2. After the mechanism collects all the bids it does the following:

   2.1. Computes the allocation using the allocation function.

   2.2. Computes the payments P_i for each agent A_i using the payment function.

   2.3. Sends P_i to each A_i.

3. Each agent receives its payment and evaluates its utility.

After receiving the payment each agent evaluates its utility and decides on the bid values for the next task. This mechanism preserves the truthfulness property in each round.

3. The Antisocial Strategy

In this section we design an antisocial strategy for the agents participating in the Modified MinWork mechanism. First we present a formal characterization of agent's antisocial behavior and then present the proposed antisocial strategy.

3.1 Background

The mechanism design approach considers the scenario where self-interested agents participate in the mechanism and there is sufficient motivation for the agents to report their true values. This is the case of truthful mechanisms in which an agent maximizes its utility by reporting its true value. An agent might misreport its type in order to inflict losses on the other agents. An agent experiences a relative loss when its payment is reduced as compared to the payment it would have received when all agents have reported their true valuations. An agent incurs an absolute loss when the payment is less than its true valuation. We denote the relative loss by RL in the subsequent text. The loss mentioned in the subsequent text is the relative loss (RL) unless

mentioned otherwise. Since the actual amount of loss varies depending on the size of the task, it is more meaningful to represent the relative loss in percentage. The relative loss (RL) is calculated as follows:

$$RL = \frac{P_i^T - P_i}{P_i} * 100 \qquad (3)$$

Where P_i^T is the payment received by the best agent A_i when all agents, including A_i, report their true values; and P_i is the payment received by the best agent A_i when one of the agents is antisocial.

An agent might accept small losses in order to inflict greater relative losses to the other agents, i.e. an agent will try to maximize the difference between its utility and the utility of the other agents. The MMW mechanism can also be viewed as running a Vickrey auction for each task. Since in the Vickrey auction the payment to the winner depends on the bid of the second best agent for that auction, an antisocial agent can reduce the profit of the winner (i.e. induce a loss) by bidding values close to the winner's bid (assuming that the antisocial agent knows the bids of the winner). To formalize the study of antisocial behavior in Vickrey auctions Brandt and Weiss [3] introduced a parameter called the *derogation rate* of an agent. The derogation rate $d_i \in [0, 1]$ can be defined as the degree of antisocial behavior of an agent A_i, i.e. d_i quantifies whether the agent gives preference to maximizing its profit or to inflict losses to the other agents. A regular agent has $d_i = 0$ and a purely destructive agent has $d_i = 1$. A balanced agent has $d_i = 0.5$ i.e. it gives equal weight to its utility and others' losses. The payoff of the agent depends on the derogation rate of the agent:

$$payoff_i = (1 - d_i)U_i - d_i \sum_{i' \neq i} U_{i'} \qquad (4)$$

Every agent tries to maximize its payoff. An antisocial agent will adjust its bid in such a manner that it is able to reduce the utility of the winner. To be able to adjust its bid an antisocial agent has to infer the actual valuation (or bid) of the winner. This is the idea that will be used to design the proposed antisocial strategy for MMW. This work extends the work in [3] which considered the antisocial behavior in repeated Vickrey auctions. In the repeated Vickrey auctions case the same item is auctioned in each round. In our case we have different items (tasks) which are related in terms of their execution times.

In the proposed antisocial strategy, the antisocial agent bids according to its derogation rate and makes decisions based on whether it was allocated the last task or not. The strategy exploits the fact that the agents (machines) are related and so the valuation of the other agents is also proportional to the task requirement (size). The agent starts by bidding its true valuation and if it loses, it reduces its bid

step by step so that it can reach the bid of the best agent. The amount by which the antisocial agent A_i reduces its bid in every step is characterized by the step down percentage, s_i. When the antisocial agent's bid is lower than the bid of the best agent, the antisocial agent wins and receives payment equal to the best agent's true value. The antisocial agent then decides its next bid such that it can inflict a relative loss to the best agent, taking into consideration its derogation rate and the bid of the best agent. Then the antisocial agent keeps bidding with the same information for the subsequent tasks inflicting losses to the best agent.

3.2 Antisocial Strategy

In describing the proposed antisocial strategy we use the following notations:

$Priceknown$ - boolean variable indicating whether A_i knows the price paid for a task in the last round (it also indicates if A_i won in the last round);

$PGreaterThanV$ - boolean variable indicating if the price received in the last round was greater than the valuation;

$DecreaseBid$ - boolean variable to control execution, it indicates if the bid is to be decreased;

$t_j = t_j^i$ - time required by antisocial agent A_i to execute the current task T_j (for simplicity we denote t_j^i by t_j since the strategy is followed by agent A_i);

$t_{j-1} = t_{j-1}^i$ - true value of agent A_i for previous task T_{j-1} (i is implicit in t_{j-1});

P_i - payment received by agent A_i in the last round (if it won);

$b_{j-1} = b_{j-1}^i$ - bid placed by agent A_i for the previous task, which might be different from the current bid (i is implicit in b_{j-1});

ϵ - a chosen small quantity;

d_i - derogation rate of the antisocial agent A_i;

s_i - percentage decrease of antisocial agent A_i's bid;

The proposed antisocial strategy is presented in Figure 1. An antisocial agent following this strategy can be assumed to be in one of the six stages at a particular point in time. An agent A_i starts in Stage 0 where it bids its true valuation. If the agent wins in Stage 0, which suggests that the valuation of this agent is the lowest (and thus also comes to know the valuation of the second best agent), it transitions to Stage 1 and bids higher than its true valuation. The bid in Stage 1 depends on the derogation rate of the agent as well

as the bid of the second best agent. The antisocial agent bids higher than its true valuation. The agent keeps bidding according to Stage 1 while it wins. If it loses it moves to Stage 2. Since the antisocial agent has lost, it does not receive the payment and so it does not have the knowledge

```
Priceknown ← false;
PGreaterThanV ← false;
DecreaseBid ← false;
j ← 1;
 Stage 0: bⱼ ← tⱼ;
        Aᵢ bids bⱼ;
        if (Aᵢ wins)
            then Pᵢ ← price;
                Priceknown ← true;
                PGreaterThanV ← true;
            else DecreaseBid ← true;
do
    j ← j + 1;
    Stage 1: if (Priceknown and PGreaterThanV)
```

$$\textbf{then } b_j \leftarrow t_{j-1} + d_i \left(\frac{t_j}{t_{j-1}} * P_i - t_j \right);$$

```
                Aᵢ bids bⱼ;
                if (Aᵢ loses)
                    then Priceknown ← false;
    Stage 2: if (not Priceknown and PGreaterThanV)
```

$$\textbf{then } b_j \leftarrow t_j +$$
$$d_i \left(\frac{t_j}{t_{j-1}} * \left(\frac{b_{j-1} - t_{j-1}}{d_i} + t_{j-1} \right) - t_j \right);$$

```
                Aᵢ bids bⱼ;
                if (Aᵢ wins)
                    then Priceknown ← true;
                        Pᵢ ← price;
                    else DecreaseBid ← true;
                        Priceknown ← false;
    Stage 3: if (DecrBid)
```

$$\textbf{then } b_j \leftarrow t_j * \left(\frac{b_{j-1}}{t_{j-1}} - s_i \right);$$

```
                Aᵢ bids bⱼ;
                if (Aᵢ wins)
                    then Priceknown ← true;
                        Pᵢ ← price;
                        PGreaterThanV ← false;
                        DecreaseBid ← false;
    Stage 4: if (Priceknown and not PGreaterThanV)
```

$$\textbf{then } b_j \leftarrow t_j - d_i \left(t_j - P_i * \frac{t_j}{t_{j-1}} \right) + \epsilon;$$

```
                Aᵢ bids bⱼ;
                if (Aᵢ wins)
                    then Pᵢ ← price;
                    else Priceknown ← false;
    Stage 5: if (not Priceknown and not PGreaterThanV)
```

$$\textbf{then } b_j \leftarrow \epsilon + t_j -$$
$$d_i \left(t_j - \frac{t_j}{t_{j-1}} * \left(\frac{b_{j-1} - t_{j-1} - \epsilon}{d_i} + t_{j-1} \right) \right);$$

```
                Aᵢ bids bⱼ;
                if (Aᵢ wins)
                    then Pᵢ ← price;
                        Priceknown ← true;
                    else Priceknown ← false;
while(j < m);
```

Figure 1. The antisocial strategy

of the the winning agent's bid. In Stage 2 it calculates the bid of the second best agent using the information about the second best agent's valuation from previous bid, and then bids according to that calculated value and its own derogation rate. If the agent wins in Stage 2, it returns to Stage 1 and bids according to the strategy of Stage 1. If it loses in Stage 2, it goes to Stage 3. If the agent loses in Stage 0 then it also transitions to Stage 3. In Stage 3, the antisocial agent reduces its bid by a small value anticipating to win and thus gets information about the bid of the agent with lower valuation. The agent remains in Stage 3 and keeps lowering the bid by a small percentage until its own bid is below the bid of the agent with the lowest valuation. When the agent wins, it goes to Stage 4. Since now it knows the valuation of the best agent, it bids lower than its own true valuation (but higher than the valuation of the best agent) in order to inflict loss on that agent. If the agent wins in Stage 4 and the payment it receives is less than its true valuation, it remains in the same stage. This means that the antisocial agent had bid less than the agent with the lowest valuation and since now the antisocial agent comes to know the bid of the best agent, it raises the bid (by bidding again according to strategy in Stage 4). If the agent wins in Stage 4, but the payment it receives is greater than its true valuation, it goes to Stage 1 and bids according to the strategy of Stage 1 in subsequent bids. This means that the valuation of the competing agent is greater than that of the antisocial agent and so the antisocial agent bids according to the strategy in Stage 1 thus bidding more than its true valuation. If the agent loses in Stage 4, it moves to Stage 5. In Stage 5, the agent calculates the bid of the agent with the lowest valuation using the information from previous bid and then bids accordingly. If the agent wins in Stage 5, it moves to Stage 4 or Stage 1 depending on whether the price it receives is less than or greater than its own true valuation respectively. If the agent looses in Stage 5, it remains in that stage where it possibly inflicts losses on the agent with the lowest valuation. The actual loss to the winner agent depends on the derogation rate of the antisocial agent and presence of another agent with valuation in between that of the antisocial agent and the winner.

4. Experimental results

To validate and analyze the proposed antisocial strategy, we developed a discrete event simulator which facilitates the simulation of the mechanism and the strategy. We simulated the antisocial strategy considering different values of the parameters in order to examine their effect on the agent losses. The simulation environment and the results are presented in this section. In the following we denote by π_i the position of an antisocial agent A_i in the sequence of agents sorted in decreasing order of their valuations. For example

Table 1. Summary of parameter values used in the simulation

Parameter	Range/Value	Parameters kept constant
Number of agents (n)	16	-
ϵ	0.01% of task size	-
Bid decrease (s_i)	5%	-
True values (t_j^i)	normal($0 \ldots 2$)	-
Antisocial position (π_i)	$1 \ldots 16$	t_j^i
Derogation rate (s_i)	$0 \ldots 1$	t_j^i, π_i
Task size (r_j)	uniform($0 \ldots 100$)	t_j^i, π_i, d_i
Number of tasks (m)	10000	t_j^i, π_i, d_i

$\pi_i = 1$ means the agent with the lowest true value, $\pi_i = 2$ means agent with the second lowest true value and so on.

4.1 Simulation Environment

The simulation was run with different combinations of parameters (presented in Table 1) to study their effect on the agents' losses. The parameters that were varied were the true values of the agents, the antisocial agent and the derogation rate. For a particular set of parameters, the mechanism allocates the tasks considering a normal scenario (no antisocial agent) and then considering the presence of one antisocial agent. The difference in the outcomes is then recorded and analyzed.

We simulated the strategy by considering a task scheduling scenario with 16 agents. The value of ϵ (a chosen infinitesimal quantity) was set to 0.01% of the task size (taken relative to task size to remove scaling errors). Also the bid percentage decrease s_i was set to 5%. This parameter decides how much to decrease the bid when an agent loses, in order to become closer to the best agent's bid. For each simulation experiment one set of true values of the agents were generated according to the normal distribution. For a particular set of true values, the simulation was run for different antisocial positions (i.e. making the agent with the lowest true value as antisocial, then the second lowest as antisocial and so on). For a particular set of true values and antisocial position, the derogation rates for that antisocial agent were varied. For a particular set of true values, antisocial position and derogation rate the simulation was run for 10^5 tasks of different sizes. The task sizes were generated according to the uniform distribution. The allocation of tasks was done according to MMW mechanism presented in Section 2.5.

For example, for a particular set of bids, the best agent is made antisocial, the experiment will be run for derogation rates of this agent from 0 to 1 (increasing derogation rate by 0.1 in each run). After that the second best agent is made antisocial and the experiment is repeated for dero-

gation rates varying from 0 to 1 and so on. This way all 16 agents were made antisocial one by one and then the set of bids is changed and the experiment repeated. For each run the allocation, the payment made, the position of antisocial agent and related data is recorded considering both scenarios (no antisocial agent and one antisocial agent is present). This data is used to calculate the percentage relative loss (RL) the antisocial agent was able to inflict on the best agent. For a particular set of true values, antisocial position and derogation rate, maximum and average relative loss (in %) is recorded. For a particular antisocial agent position, the maximum and average (over varying derogation rate) of relative loss that agent is able to inflict is recorded. For a particular set of bids also the maximum and average percent relative loss is recorded.

4.2 Results

The following objectives were achieved by running the simulation:

(i) Verify the correctness of the proposed antisocial strategy.

(ii) Analyze the effect of derogation rate on the amount of loss inflicted.

(iii) Analyze the effect of antisocial agent position on the amount of loss inflicted.

(iv) Analyze the effect of true value on the amount of loss inflicted.

Figure 2 shows the relative losses inflicted by an antisocial agent for the first one hundred rounds that correspond to the first one hundred allocated tasks. The agent starts bidding from its true valuation and decreases its bid (thus increasing the relative loss to the best agent) until it is allocated the task and thus comes to know the valuation of the best agent. According to the strategy, after this antisocial agent reaches the stable phase (here after task 10 is allocated) it keeps bidding according to its derogation rate, inflicting losses to the best agent. This continues until all tasks are finished. The maximum loss inflicted is about 47% in this case.

In Figure 3 we present the maximum relative loss inflicted to the best agent in two cases in which agent at position 2 and 4 are antisocial. It can be seen that if we consider the second agent as antisocial, then by reducing its bid, it is always able to inflict losses. The loss percent grows with the derogation rate of the agent. If an agent other than the second best agent is antisocial, it is able to inflict losses only when its bid is less than the second best agent's bid. So even if the agent is able to infer the valuation of the best agent, it is not able to always inflict losses. As can be seen from the figure, the antisocial agent in position 4 ($\pi_i = 4$) is able to inflict losses only after its derogation rate is 0.6. Before that even if it is bidding less than its true valuation, it is not
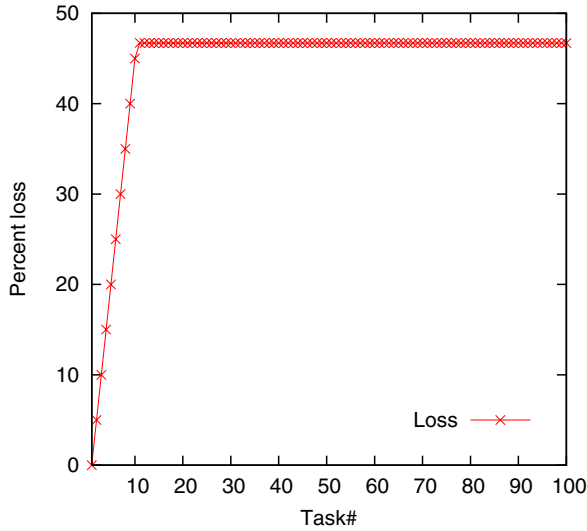
IEEE
COMPUTER
SOCIETY

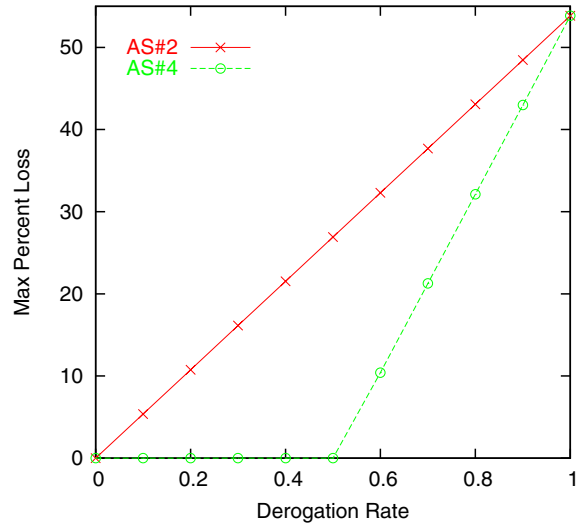**Figure 2. Relative loss** RL **(in percent) for each task**

Figure 3. Maximum relative loss inflicted vs. derogation rate (antisocial agents at position 2 and 4)

able to inflict any losses, since its bid is higher than the bid of second best agent, which decides the payment to the best agent. However all the agents inflict maximum losses when their derogation rate is 1, i.e. they are purely destructive.

Considering Figure 4, it can be seen that the maximum relative loss any agent can inflict on the best agent is equal to the percentage difference between the valuations of the best agent and the second best agent (in this case 54%). Also the average loss an agent can inflict (with different derogation rates) decreases as the position of antisocial agent increases (i.e. the higher the antisocial agent position, the lower the average loss). This is due to the fact that the antisocial agent is able to inflict losses only when its derogation rate is sufficiently high thus making the bid of the antisocial agent less than the bid of the second best agent (but greater than the best agent). The agents having a greater difference between their true valuations and the valuation of the best agent are able to inflict losses only when their derogation rate is sufficiently high and thus the average loss percentage is low.

5. Conclusion

As can be seen from the results presented in this paper, the presence of an antisocial agent in the task scheduling scenario can greatly affect the profit of the other agents. This is due to the second price policy followed by the mechanism we consider. If an agent wants to inflict losses on the other agents or wants to reduce its own losses due to the presence of other antisocial agents, the antisocial strat-
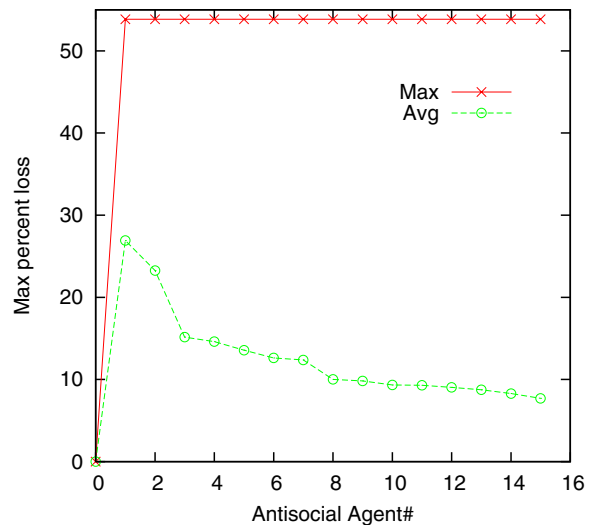

Figure 4. Maximum and average relative loss inflicted vs. antisocial agent number

egy presented in this paper should be applied. We analyzed the effect of different parameters on the antisocial strategy. In particular agents with high derogation rates inflict more losses on the other agents. Also the amount of loss depends on the difference in true values of the agent with the lowest valuation, the agent with the second lowest valuation and the antisocial agent (if different from above two). The

amount of loss that can be inflicted also depends on the number of agents having the true values lying in between the true values of the best agent and the antisocial agent. In future research we will consider the presence of more than one antisocial agent and study its effect on agents' utilities.

References

[1] A. Archer and E. Tardos. Truthful mechanism for one-parameter agents. In *Proc. of the 42nd IEEE Symp. on Foundations of Computer Science*, pages 482–491, October 2001.

[2] A. Archer and E. Tardos. Frugal path mechanisms. In *Proc. of the 13th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 991–999, January 2002.

[3] F. Brandt and G. Weiß. Antisocial agents and Vickrey auctions. In *Proc. of the 8th Workshop on Agent Theories, Architectures and Languages*, pages 335–347, August 2001.

[4] E. Clarke. Multipart pricing of public goods. *Public Choice*, 8:17–33, 1971.

[5] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A bgp-based mechanism for lowest-cost routing. In *Proc. of the 21st ACM Symp. on Principles of Distributed Computing*, pages 173–182, July 2002.

[6] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63(1):21–41, August 2001.

[7] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proc. of the 6th ACM Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 1–13, September 2002.

[8] A. V. Goldberg and J. D. Hartline. Collusion-resistant mechanisms for single-parameter agents. Technical Report MSR-TR-2004-40, Microsoft Research - Silicon Valley, September 2004.

[9] D. Grosu and A. T. Chronopoulos. Algorithmic mechanism design for load balancing in distributed systems. *IEEE Trans. Systems, Man and Cybernetics, Part B*, 34(1):77–84, February 2004.

[10] T. Groves. Incentive in teams. *Econometrica*, 41(4):617–631, 1973.

[11] R. Karp, E. Koutsoupias, C. H. Papadimitriou, and S. Shenker. Optimization problems in congestion control. In *Proc. of the 41st IEEE Symp. on Foundations of Computer Science*, pages 66–74, November 2000.

[12] N. Nisan, S. London, O. Regev, and N. Camiel. Globally distributed computation over Internet - The POPCORN project. In *Proc. of the 18th IEEE International Conference on Distributed Computing Systems*, pages 592–601, May 1998.

[13] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behaviour*, 35(1/2):166–196, April 2001.

[14] M. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, Cambridge, Mass., 1994.

[15] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Prentice Hall, Upper Saddle River, NJ, 2002.

[16] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, March 1961.

[17] W. E. Walsh, M. P. Wellman, P. R. Wurman, and J. K. MacKie-Mason. Some economics of market-based distributed scheduling. In *Proc. of the 18th IEEE International Conference on Distributed Computing Systems*, pages 612–621, May 1998.

[18] R. Wolski, J. S. Plank, T. Bryan, and J. Brevik. G-commerce: market formulations controlling resource allocation on the computational grid. In *Proc. of the 15th IEEE International Parallel and Distributed Processing Symposium*, April 2001.