From: AAAI Technical Report WS-02-06. Compilation copyright © 2002, AAAI (www.aaai.org). All rights reserved.

Miscomputing Ratio: The Social Cost of Selfish Computing

Kate Larson and Tuomas Sandholm
Computer Science Department
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
{klarson,sandholm}@cs.cmu.edu

Abstract

Auctions are useful mechanism for allocating items (goods, tasks, resources, etc.) in multiagent systems. The bulk of auction theory assumes that the bidders' valuations for items are given *a priori*. In many applications, however, the bidders need to expend significant effort to determine their valuations. In this paper we analyze computational bidder agents that can refine their valuations (own and others') using computation. We introduce a way of measuring the negative impact of agents choosing computing strategies selfishly. Our *miscomputing ratio* isolates the effect of selfish computing from that of selfish bidding. We show that under both limited computing and costly computing, the outcome can be arbitrarily far worse than in the case where computations are coordinated. However, under reasonable assumptions on how limited computing changes valuations, bounds can be obtained. Finally, we show that by carefully designing computing cost functions, it is possible to provide appropriate incentives for bidders to choose computing policies that result in the optimal social welfare.

Introduction

Auctions are useful mechanisms for allocating items (goods, tasks, resources, etc.) in multiagent systems. The bulk of auction theory assumes that the bidders' valuations for items are given *a priori*. In many applications, however, the bidders need to expend significant effort to determine their valuations. This is the case, for example, when the bidders can gather information (Perisco 2000) or when the bidders have the pertinent information in hand, but evaluating it is complex. There are a host of applications of the latter that are closely related to computer science and AI questions. For example, when a carrier company bids for a transportation task, evaluating the task requires solving the carrier's intractable vehicle routing problem (Sandholm 1993). As another example, when a subcontractor bids for a manufacturing job, evaluating it requires computing the subcontractor's manufacturing plan.

A normative deliberation control model of how additional work (e.g., computing) refines valuations was recently introduced (Larson & Sandholm 2001c; 2001b). The authors analyzed auctions strategically, where each agent's strategy included both computing and bidding. They found that for certain auctions, properties such as incentive compatibility cease to hold if agents explicitly deliberate to determine valuations. Instead agents strategize and counterspeculate, sometimes using computing to (partially) determine opponents' valuations. It was conjectured that such strategic computing may lead to inefficient outcomes.

In this paper we introduce a way of measuring the negative impact of agents choosing computing strategies selfishly. Our *miscomputing ratio* isolates the effect of selfish computing from that of selfish bidding. We show that under both limited computing and costly computing, the outcome can be arbitrarily far worse than in the case where computations are coordinated. However, under reasonable assumptions on how limited computing changes valuations, bounds can be obtained. Finally, we show that by carefully designing computing cost functions, it is possible to provide appropriate incentives for bidders to choose computing policies that result in the optimal social welfare.

The paper is organized as follows. The next section describes the auction model and deliberation model. The following section discusses why Pareto efficiency is not necessarily a good way of measuring the impact of restricted computing on the outcome of the auction. This is followed by the introduction of our miscomputing ratio, and the results we derive for it. We conclude with related work and a summary of the paper.

The Model

In this section we specify our model. We first review game-theoretic solution concepts, then auctions, and finally present the model of deliberation control.

Copyright © 2002, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

Concepts from Game Theory

A game has a set of agents and a set of outcomes O. Each agent has a set of strategies from which it chooses a strategy to use. A strategy is a contingency plan that determines what action the agent will take at any given point in the game. A *strategy profile*, $s = (s_1, \ldots, s_n)$, is a vector specifying one strategy for each player i in the game. We use the notation $s = (s_i, s_{-i})$ to denote a strategy profile where agent i's strategy is s_i and $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$. The strategies in the profile determine how the game is played out, and thus determine the outcome $o(s) \in O$. Each agent i tries to choose its strategy, s_i, to as to maximize its utility, which is given by a utility function $u_i : O \mapsto \mathbb{R}$.

Noncooperative game theory is interested in finding stable points in the space of strategy profiles. These stable points are the *equilibria* of the game. There are many types of equilibria but in this paper we focus on the two most common ones: *dominant strategy equilibria* and *Nash equilibria*.

A strategy is said to be *dominant* if it is a player's strictly best strategy against any strategies that the other agents might play.

Definition 1 *Agent i's strategy s_i^* is a dominant strategy if*

$$\forall s_{-i}\ \forall s_i' \neq s_i^*\ u_i(o(s_i^*, s_{-i})) > u_i(o(s_i', s_{-i})).$$

The strategy is weakly dominant if the inequality is not strict.

If each agent's strategy in a strategy profile is the agent's dominant strategy, then the strategy profile is a *dominant strategy equilibrium.*

Agents may not always have dominant strategies and so dominant strategy equilibria do not always exist. Instead a different notion of equilibrium is often used, that of the Nash equilibrium.

Definition 2 *A strategy profile s^* is a* Nash equilibrium *if no agent has incentive to deviate from his strategy given that the other players do not deviate. Formally,*

$$\forall i\ \forall s_i'\ u_i(o(s_i^*, s_{-i}^*)) \geq u_i(o(s_i', s_{-i}^*)).$$

The Nash equilibrium is strict *if the inequality is strict for each agent.*

In this paper, whenever we measure outcomes, we measure them from the perspective of the bidders in the auction, not caring about the auctioneer (who is not a strategic agent in our model). One common measure for comparing outcomes is Pareto efficiency. It is a desirable measure in the sense that it does not require cardinal utility comparisons across agents.

Definition 3 *An outcome o is* Pareto efficient *if there exists no other outcome o' such that some agent has higher utility in o' than in o, and no agent has lower utility. Formally, $\nexists o'$ s.t. $[\forall i, u_i(o') \geq u_i(o)$ and $\exists i\ u_i(o') > u_i(o)]$.*

Another measure that is commonly used is *social welfare.* It often allows prioritizing one Pareto efficient outcome over another, but it does require cardinal utility comparison across agents.

Definition 4 *The* social welfare *of outcome $o \in O$ is $SW(o) = \sum_i u_i(o)$.*

Equilibrium play does not always optimize social welfare. A classic example of this is the Prisoner's Dilemma game.

The definitions given above were for general utility functions. However, in this paper, as is standard when discussing auctions, we assume that the agents' utility functions are *quasi-linear.* That is, the utility of agent i, u_i, is of the form $u_i = v_i - p_i$ where v_i is the amount that the agent values the item up for auction and p_i is the amount that it pays for the item. If agent i does not win the auction, then $u_i = 0$.

Auctions

In this paper we consider auctions where one good is being sold. There are numerous auction mechanisms, but in this paper we focus on the Vickrey auction. In a Vickrey auction (aka. second-price sealed-bid auction), one good is being sold, each bidder can submit one sealed bid, the highest bidder wins, but only pays the price of the second-highest bid. The desirable feature of this mechanism is that if a bidder knows its private valuation for the good, the bidder's (weakly) dominant strategy is to bid that valuation (rather than strategically under- or over-bidding). We chose to study the Vickrey auction because it has this desirable property in the classic literature, but ceases to have this property when the bidder agents do not know their own valuations, but rather have the option of investing computation to determine them. In our model, the agent's valuations are independent of each other as in most of the literature, but we deviate in that our agents do not know their own valuation *a priori.*

Normative Model of Deliberation

In order to participate in an auction, agents need to be able to have a valuation for the items being sold. The question is: How are these valuations obtained? In this paper we focus on settings where agents do not simply know their own valuations. Rather they have to allocate computational resources to compute the valuations.

If agents know their own valuations (or are able to determine them with ease) they can execute the equilibrium bidding strategies for rational agents.

However, agents often have restrictions on their capabilities for determining the valuations. In this paper we are interested in settings where agents have to *compute* to determine valuations. Settings where the value of an item depends on how it is used often has this property. For example, valuation determination may involve solving optimization problems that provide a solution as to how the items in the auction can be used once obtained. However, many optimization problems, such as scheduling, are NP-complete. It may not be feasible to optimally solve the valuation problems. Instead, some form of approximation must be used. In this paper we assume that agents have *anytime algorithms* (Boddy & Dean 1994). The defining property of an anytime algorithm is that it can be stopped at any point in time to provide a solution to the problem, and the quality of the solution improves as more time is allocated to the problem. This allows a tradeoff to be made between solution quality and time spent on computing. Since the amount of time an agent can use to compute valuations is limited by deadlines or cost, the agents must make tradeoffs in how to determine their valuations. Alone, anytime algorithms do not provide a complete solution. Instead, they are paired with a meta-level control procedure that determines how long to run an anytime algorithm, and when to stop and act with the solution obtained. In this paper we assume that agents have a meta-level control procedure in the form of performance profile trees, based on work in (Larson & Sandholm 2001a).

There is a performance profile tree for each valuation problem (one valuation problem per agent). Figure 1 presents one such tree. The trees are obtained from statistics collected from previous runs of an algorithm on the valuation problem. The tree describes how deliberation (computation) changes the solution to the valuation problem. Each agent uses this information to decide how to allocate its computing resources at each step in the process, based on results of its computing so far.

The trees capture uncertainty that stems from both randomized algorithms and variation of performance on different problem instances. There are two different types of nodes in the performance profile tree, solution nodes and random nodes. Each solution node stores the solution that the algorithm has computed given a certain amount of computation so far. Random nodes occur whenever a random number is used to chart the path of the algorithm run. The edges in the tree are labeled with the probability that after one more step of computation, the solution returned will be the node found by following the edge.

Agents use the performance profile trees to help in making decisions about how to use their computational resources. As agents allocate computational time to an algorithm, the solutions returned move the agent from parent to child in the tree. The performance profile trees provide information about how the solution is likely to improve with future computation. In particular, if an agent has reached a solution corresponding to a node in the tree, then the agent need only consider solutions in the subtree rooted at the node. The probability of obtaining a solution v', given that the agent has reached a node with solution v, is equal to the product of the probabilities of the edges connecting node with solution v to v'.

There are two different types of performance profiles: *stochastic* and *deterministic*. A *stochastic performance profile* models uncertainty as to what results future computing will bring. At least one node in the tree has multiple children. The uncertainty can come from variation in performance on different problem instances or from the use of randomized algorithms. A *deterministic performance profile* is the special case where the algorithm's performance can be projected with certainty (i.e., the tree is a branch). With a deterministic performance profile, an agent can determine what the solution will be after any number of computing steps devoted to the problem—before the agent conducts any computation. Even though the agent knows what solution it can obtain, it must still compute in order to obtain it. Figure 1 is an example of a stochastic performance profile tree.

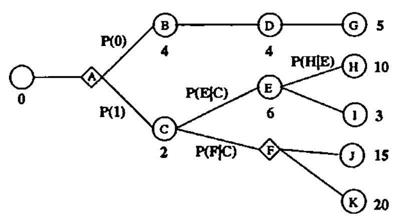

Figure 1: *An agent's stochastic performance profile tree for a valuation problem. The diamond shaped nodes are random nodes and the round nodes are solution nodes. At random node A, the probability that the random number will be 0 is $P(0)$, and the probability that the random number will be 1 is $P(1)$. AT solution node E, the edges are labeled with the probability of reaching each child, given that node E was reached.*

The performance profile tree is a fully normative model for deliberation control which is required for game theoretic analysis. It also allows optimal conditioning on many parameters, including results of execution so far and on the actual problem instance.

In the rest of the paper we make the assumption that all performance profiles are common knowledge. This means, that all agents know what all performance profiles look like, and they know that all the agents know. Agents are allowed to compute on each others' problems. We do not assume that agents know how their opponents are computing.

Strategic Computing and Bidding

We consider two models of computing. In one of them, computing is free, but there is a deadline for each agent when that agent has to stop computing. In the other model, the computations do not have deadlines, but each agent has to pay for the cycles it consumes. Let T be the time when the auction closes. After that the agent cannot bid or compute valuations. In the model of limited computing, each agent has T free computing cycles to use. In the model of costly computing, each agent can consume as many cycles per real-time unit as it wants, but has to pay a computing cost $c_i(\cdot)$.

At every step of the game, each agent can take a computing action (the agent can also skip taking a computing action). Taking a computing action means allocating one step of computing on one's own valuation problem or on one of the other agents' valuations problems (so as to obtain information about their valuations, which the agent can use to bid more strategically to benefit itself). We say that an agent uses *strong strategic computing* if it allocates some of its computing cycles on others' valuation problems.

At the deadline T, each agent submits one sealed bid to the Vickrey auction. This bid is the agent's bidding strategy. The amount an agent bids depends on the solutions it has obtained for its (and others') valuation problems through computing.

It has been shown in earlier work that the model of computing (costly or limited) has a significant impact on what strategies agents may use:

Theorem 1 *Assume that agents have free but limited computing. Then, in a Vickrey auction, the bidders have (weakly) dominant strategies where they only compute on their own valuation problems (Larson & Sandholm 2001b).*

Theorem 2 *Assume that agents have unlimited but costly computing. Then, in a Vickrey auction, strong strategic computation can occur in strict Nash equilibrium (Larson & Sandholm 2001c).*

The Social Cost of Selfish Computing

Now, a natural question to ask is whether the cost or limit on computing resources results in a loss of efficiency. However, efficiency is hard to compare in such settings. The Vickrey auction is efficient in the sense that it always allocates the item to the bidder with the highest valuation. However, an agent who *might* have been able to obtain the highest valuation via computing, may have used its limited computing on a different problem, thus causing a different agent to have the highest valuation and win the auction. This come is still efficient *given how agents computed*, but it overlooks the computational issues in an unsatisfying way. This suggests that Pareto efficiency may not always be the right measure to use in the context of computationally bounded agents. Is there an alternative measure?

Instead of looking at efficiency, we propose to use social welfare as the measure. We want to know how letting agents freely choose their own computing strategies impacts the social welfare of the set of all bidders. In particular, we compare the highest achievable social welfare to the lowest social welfare achievable in any Nash equilibrium.

When we determine the highest achievable social welfare we optimistically assume that there is a global controller who imposes each agent's computing strategy (so as to maximize social welfare). The controller has full information about all performance profiles, deadlines, cost functions, and intermediate results of computing, and given this information, specifies exactly how each agent must use its computational resources. In the bidding stage agents are free to bid as they wish, but their goal is still to maximize their own utility, and so they bid truthfully in the Vickrey auction, given the valuations they have obtained under the enforced computing policy.

Definition 5 *Let o^* be the outcome that is reached if the global controller dictates computing policies to all agents, and agents are free to bid in the Vickrey auction.*

On the other extreme, we are interested in what happens when agents are free to choose to follow any computing and bidding strategy. Let NashEq be the set of Nash equilibria in that game. We now define what is meant by the worst-case Nash equilibrium.

Definition 6 *The worst case Nash equilibrium is*

$$NE = \operatorname*{argmin}_{s \in \text{NashEq}} SW(o(s)).$$

We use the following ratio to see how much letting agents choose their own computing strategies reduces the social welfare.

Definition 7 *The* miscomputing ratio *is*

$$R = \frac{SW(o^*)}{SW(o(NE))}.$$

This ratio isolates the impact of selfish computing from the traditional strategic bidding behavior in auctions. This is because in both the coordinated and uncoordinated scenario, the agents bid based on self-interest.

Results

In this next section we present our results in terms of the miscomputing ratio. The first subsection discusses the general case with limited computing. The next subsection studies how the ratio can be improved when the analyzer has more knowledge.

The following subsection studies the general case with costly computing. The final subsection shows how the costs can be adjusted to increase social welfare.

General Case with Limited Computing

It turns out that with limited computing, the miscomputing ratio can be arbitrarily bad.

Proposition 1 *Assume there are n bidders in a Vickrey auction, each bidder has free but limited computing, and the auction closes at time T. Then, the miscomputing ratio R can be infinity.*

Proof: Assume that all agents have deterministic performance profiles. Each agent has a dominant strategy which is to deliberate only on its own valuation problem until the deadline and to submit a bid equal to the valuation that it has obtained. That is, agent i submits a bid of $v_i(T)$. Without loss of generality, assume that $v_1(T) \geq v_2(T) \geq v_j(T)$ for all $j \neq 1, 2$. In equilibrium, agent 1 will win the auction and pay an amount of $v_2(T)$. Therefore, agent 1's utility is $u_1 = v_1(T) - v_2(T)$. Set $u_1 = \epsilon$. The utility for all other agents is $u_i = 0$ for $i \neq 1$. Therefore,

$$SW(o(\mathrm{NE})) = \sum_{j=1}^{n} u_j = \epsilon.$$

In order to maximize social welfare, the global controller would prohibit all agents expect for agent 1 to deliberate. Agent 1 would compute on its valuation problem until time T and submit a bid of $v_1(T)$ while all other agents would submit a bid of 0. Agent 1 would win the item and pay an amount of 0. The utility for agent 1 is $u_1 = v_1(T) - 0 = v_1(T)$, while $u_i = 0$ for all $i \neq 1$. Therefore

$$SW(o^*) = \sum_{j=1}^{n} u_j = v_1.$$

The ratio, R, is

$$R = \frac{SW(o^*)}{SW(o(\mathrm{NE}))} = \frac{v_1(T)}{\epsilon}.$$

As $\epsilon \to 0$ (that is, as the difference between the highest and second highest valuations decreases), $R \to \infty$. □

This is a negative result. Allowing agents to choose their computing strategies leads to an outcome that can be arbitrarily far from optimal.

Bounding the Miscomputing Ratio Under Limited Computing

However, in many situations the miscomputing ratio will not be unbounded. Even if the performance profiles are stochastic, as long as the difference between the highest computed valuation and the second highest computed valuation is "large enough", then the ratio will not be unbounded. Let k be the difference between the highest possible computed valuation and the second highest possible computed valuation. That is

$$k = \min[\max_i \max_{v_i(T)} v_i(T) - \max_{j \neq i} \max_{v_j(T)} v_j(T)]^1$$

under the constraint that $v_i(T) > v_j(T)$. The amount k is equal to the lowest possible social welfare obtainable if agents compute in a selfish manner. If guarantees on the size of k can be made by the restriction of performance profile trees then the miscomputing ratio can be made finite.

Proposition 2 *Let*

$$k = \min[\max_i \max_{v_i(T)} v_i(T) - \max_{j \neq i} \max_{v_J(T)} v_j(T)]$$

for all i, j and all possible values of $v_i(T)$ and $v_j(T)$ under the constraint that $v_i(T) > v_j(T)$. Then the miscomputing ratio is

$$R \leq \frac{\max_i \max v_i(T)}{k}.$$

General Case with Costly Computing

If agents have costly unlimited computing, then they no longer necessarily have dominant strategies in the Vickrey auction (see Theorem 2). Instead, what they do depends on what strategies the other agents choose. When placing bids, agents no longer directly bid the valuation that they have computed. Instead, they shave the bids downwards.

By constructing appropriate cost functions, it turns out to be possible to emulate the situation where agents have free computing but are limited by deadlines. Therefore it is not surprising that under certain circumstances the ratio of the maximum social welfare to the social welfare obtained from the worst Nash equilibrium can be unbounded.

Proposition 3 *Consider a Vickrey auction with n bidders. Assume that each bidder i has costly, unlimited computing. Then, the miscomputing ratio R can be infinity.*

Proof: Assume that each agent i has the following cost function, $c_i(t)$;

$$c_i(t) = \begin{cases} 0 & \text{if } t \leq T; \\ \infty & \text{if } t > T. \end{cases}$$

Each agent has a dominant strategy which is to deliberate only on is own valuation problem until time T and then submit a bid of $v_i(T)$. That is, each agent behaves as though they have free but limited computing resources with a deadline at time T. Like in the proof for the free but limited agents,

[1] If the performance profiles are stochastic there may be multiple valuations that could be computed for each agent.

assume that the difference between the highest and second highest bids is ϵ and, without loss of generality, assume that the highest valuation is $v_1(T)$. Then

$$R = \frac{v_1(T)}{\epsilon}$$

and as $\epsilon \to 0$, $R \to \infty$. $\square$

Adjusting the Computing Cost to Increase Social Welfare

Prior literature has shown that in Vickrey auctions, computationally limited agents have no incentive to use strong strategic computing (i.e., they do not counterspeculate each other) while agents with costly computing do (Larson & Sandholm 2001c). This suggests that if there is a system designer who can control how the agents' computational capabilities are restricted, the designer should rather impose limits than costs.

However, it turns out that computing costs can be adjusted so that the optimal miscomputing ratio ($R = 1$) is reached. This would mean that charging for computing is at least as desirable as imposing limits.

Proposition 4 *Computing cost functions can be used to motivate bidders to choose strategies that maximize social welfare.*

Proof: Consider the following example. Let there be 2 agents, agent 1 and agent 2, each with a *deterministic performance profile*. Assume that both agents have free but limited computing resources. Each agent has a dominant strategy, which is to deliberate on their own problem and submit a bid of $v_i(T)$. Assume that $v_1(T) > v_2(T)$. The equilibrium outcome is to award the item to agent 1 and have agent 1 pay an amount $v_2(T)$. Agent 1's utility is then $u_1 = v_1(T) - v_2(T)$ while agent 2's utility is $u_2 = 0$. To maximize social welfare the global controller would forbid agent 2 to deliberate, and thus agent 1 could get the item and need not pay anything. The maximum social welfare would be $u_1 = v_1(T)$. Therefore

$$R = \frac{v_1(T)}{v_1(T) - v_2(T)}$$

Next, consider the case where a simple cost function is introduced. Define

$$c_i(t) = \begin{cases} c \text{ if } t \leq T; \\ \infty \text{ if } t > T; \end{cases}$$

for some constant c, $0 < c \leq v_2(T) \leq v_1(T)$. Any strategy that involves deliberating on the other agent's valuation problem is dominated as the computing action incur a cost without improving the agent's overall utility. Thus, the remaining strategies are for the agents to compute only on their own valuation problem until the cost becomes too high,

	compute	no
compute	$v_1(T) - v_2(T), -c$	$v_1(T) - c, 0$
no	$0, v_2(T) - c$	0,0

Table 1: *Normal form game. Agent 1 is the row player and agent 2 is the column player. Each agent would submit a bid that is equal to its computed valuation minus the cost spent to obtain the valuation.*

or not to compute at all. The game can be represented in normal form in Table 1.

The sole Nash equilibrium is for agent 1 to compute and submit a bid of $v_1(T) - c$ and for agent 2 to not compute. The global controller trying to maximize the social welfare would force each agent to also follow those strategies. Therefore

$$R = \frac{v_1(T) - c}{v_1(T) - c} = 1.$$

In this example the constant c can be made arbitrarily close to zero. Therefore, the maximum social welfare generated by the global controller in the costly computing setting and be made arbitrarily close to the maximum social welfare obtainable if computing resources are free. $\square$

Related Research

In auctions, computational limitations have been discussed both as they pertain to bidding agents and as they pertain to running the auction (the mechanism). For bounded–rational bidding agents, Sandholm noted that under a model of costly computing, the dominant strategy property of Vickrey auctions fails to hold (Sandholm 2000). Instead, an agent's best computing action can depend on the other agents. In recent work, auction settings where agents have hard valuation problems have been studied (Larson & Sandholm 2001c; 2001b; Parkes 1999). Parkes presented auction design as a way to simplify the meta–deliberation problems of the agent, with the goal of providing incentives for the "right" agents to deliberate for the "right" amount of time (Parkes 1999). Recently Larson and Sandholm have been working on incorporating computing actions into agents' bidding strategies using a normative model of deliberation control and have focused on equilibrium analysis of different auction settings under different deliberation limitations (Larson & Sandholm 2001b; 2001c). While we borrow the deliberation model from Larson and Sandholm, this paper addresses a different question from previous work. They investigate the impact of restricted computing capabilities on agents' strategies, we look, instead, at what the impact is at a system-wide level, present a measure for comparing overhead in different settings, and ask if it is possible to place certain bounds on the overhead added by having resource-bounded agents.

There has also been recent work on computationally limited mechanisms. In particular, research has focused on the generalized Vickrey auction and has investigated ways of introducing approximate algorithms or using heuristics to compute outcomes without loosing incentive compatibility (Nisan & Ronen 2000; Kfir-Dahav, Monderer, & Tennenholtz 2000). Our work is different in that it is focused on settings where the agents are computationally limited.

Koutsoupias and Papadimitriou (Koutsoupias & Papadimitriou 1999) first proposed the concept of worst-case Nash equilibrium. This has been called the *price of anarchy* (Papadimitriou 2001). They focused on a network setting where agents must decide how much traffic to send along paths in the network. The agents did not have computational limitations. Roughgarden and Tardos studied a different model of network routing using the same measure as Koutsoupias and Papadimitriou and obtained tight bounds as to how far from the optimal outcome the agents would be, if allowed to send traffic as they wished (Roughgarden & Tardos 2000).

Conclusions

Auctions are useful mechanism for allocating items (goods, tasks, resources, etc.) in multiagent systems. The bulk of auction theory assumes that the bidders' valuations for items are given *a priori*. In many applications, however, the bidders need to expend significant effort to determine their valuations. In this paper we studied computational bidder agents that can refine their valuations (own and others') using computation. We borrowed a normative model of deliberation control for this purpose.

We focused on the Vickrey auction where bidding truthfully is a dominant strategy in the classical model. It was recently shown that this is not the case for computationally restricted agents. In this paper we introduced a way of measuring the negative impact of agents choosing computing strategies selfishly. Our *miscomputing ratio* compares the social welfare obtainable if a global controller enforces computing policies designed to maximize social welfare (but does not impose bidding strategies), to the social welfare that is obtained in the worst Nash equilibrium. This measure isolates the effect of selfish computing from that of selfish bidding.

We showed that under both limited computing and costly computing, the outcome can be arbitrarily far worse than in the case where computations are coordinated. However, under reasonable assumptions on how limited computing changes valuations, bounds can be obtained. Finally, we showed that by carefully designing computing cost functions, it is possible to provide appropriate incentives for bidders to choose computing policies that result in the optimal social welfare. This suggests (unlike earlier results) that, if there is a system designer that can choose how to restrict the agents' computing, imposing costs instead of limits may be the right approach.

Ackowledgments

This material is based upon work supported by the National Science Foundation under CAREER Award IRI-9703122, Grant IIS-9800994, and ITR IIS-0081246.

References

Boddy, M., and Dean, T. 1994. Deliberation scheduling for problem solving in time-constrained environments. *Artificial Intelligence* 67:245–285.

Kfir-Dahav, N.; Monderer, D.; and Tennenholtz, M. 2000. Mechanism design for resource bounded agents. In *Proceedings ICMAS-2000.*

Koutsoupias, E., and Papadimitriou, C. 1999. Worst-case equilibria. In *Symposium on Theoretical Aspects in Computer Science.*

Larson, K., and Sandholm, T. 2001a. Bargaining with limited computation: Deliberation equilibrium. *Artificial Intelligence* 132(2):183–217.

Larson, K., and Sandholm, T. 2001b. Computationally limited agents in auctions. In *AGENTS-01 Workshop of Agents for B2B*, 27–34.

Larson, K., and Sandholm, T. 2001c. Costly valuation computation in auctions. In *TARK VIII*, 169–182.

Nisan, N., and Ronen, A. 2000. Computationally feasible VCG mechanisms. In *Proceedings of the ACM Conference on Electronic Commerce* , 242–252.

Papadimitriou, C. 2001. Algorithms, games and the Internet. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, 749–253.

Parkes, D. C. 1999. Optimal auction design for agents with hard valuation problems. In *Agent-Mediated Electronic Commerce Workshop, IJCAI-99.*

Perisco, N. 2000. Information acquisition in auctions. *Econometrica* 68(1):135–148.

Roughgarden, T., and Tardos, É. 2000. How bad is selfish routing? In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science.*

Sandholm, T. 1993. An implementation of the contract net protocol based on marginal cost calculations. In *AAAI-1993*, 256–262.

Sandholm, T. 2000. Issues in computational Vickrey auctions. *International Journal of Electronic Commerce* 4(3):107–129.