Chapter 10

POLYNOMIAL TIME MECHANISMS FOR COLLECTIVE DECISION MAKING

Thomas C. O'Connell*
Department of Mathematics and Computer Science, Skidmore College, 815 North Broadway, Saratoga Springs, NY 12866, USA.
oconnellT@acm.org

Richard E. Stearns
Department of Computer Science, State University of New York at Albany, Albany, NY 12222, USA.
res@cs.albany.edu

Abstract We investigate the problem of designing mechanisms to control collective decisions made by self-interested autonomous agents. In particular, we examine how results in the economics literature on mechanism design apply to collective decisions involving NP-hard optimization problems. We formalize the idea of polynomial time mechanism design and investigate mechanism design for a multiagent version of MAXSAT. We prove that there exists a polynomial time mechanism for multiagent MAXSAT that guarantees the outcome to be within a factor of 1/2 of the optimal outcome. We also show that, in general, a 1/2 approximation is the best approximation possible for dominant strategy, Nash, undominated Nash and subgame perfect implementation. Our analysis highlights some of the difficulties that arise in applying results from mechanism design to computational problems. Our results suggest that we may be much less successful using approximation to overcome hardness results in multiagent settings than in traditional computational settings especially when we require game theoretic guarantees on the outcomes.

*Supported in part by National Science Foundation Grant CCR-97-34936

This is a preprint of an article that appears in *Game theory and decision theory in agent-based systems*, Parsons, S., Gmytrasiewicz, P. and Wooldridge, M. J. (eds.), Kluwer Academic Publishers (2002).

Keywords: Game Theory, Mechanism Design, Computational Complexity, Approximation Algorithms.

1. INTRODUCTION

Consider a warehouse inhabited by several robots that have different and possibly conflicting goals.[1] Each robot is concerned only with satisfying its own goals and does not care whether any of the other robots satisfy their goals. Rather than spending time negotiating with one another when a conflict arises, the robots rely on an outside arbitrator to resolve the conflict quickly and equitably. The arbitrator's only goal is that its decisions satisfy some measure of social desirability called a *social choice rule*. For example, the decision might be required to be *Pareto optimal* – no other possible decision makes one of the robots better off without making another robot worse off.

In the economics literature the arbitrator in the example above is called a *mechanism*. When a mechanism guarantees that a social choice rule is satisfied, the mechanism is said to *implement* the social choice rule. The problem of defining a mechanism to implement a particular social choice rule is known as the *mechanism design* or *implementation problem*. The field of mechanism design has recently attracted the interest of researchers in multiagent systems, particularly those studying automated negotiation (Rosenschein and Zlotkin, 1994; Sandholm, 1999). Before mechanism design can make practical contributions to multiagent system design, however, the computational issues involved must be clearly understood.

The literature on mechanism design is vast (see (Moore, 1992) and Chapter 23 of (Mas-Colell et al., 1995) for surveys). However, in (Moore, 1992), Moore points out that much of this research ignores issues of practicality. He cites two areas for potential research along these lines:

1. modeling bounded rationality on the part of the agents

2. designing mechanisms that are robust to errors in the specification of the problem.

In this paper, we address the idea of practicality in mechanism design from the point of view of theoretical computer science. We model bounded rationality by limiting the agents to polynomial time computation. Since we are interested in decisions that can be computed quickly, we also limit the mechanism to polynomial time computation. Our goal is to investigate the difficulties that arise when trying to implement NP-hard social choice rules and to determine what can be done to overcome

these difficulties. Section 2 provides a short introduction to mechanism design. Section 3 formalizes what we mean by a polynomial time mechanism. Sections 4 and 5 look at designing polynomial time mechanisms using dominant strategy and Nash equilibrium respectively for a multiagent version of MAXSAT. In multiagent MAXSAT, each agent's preferences over the set of possible outcomes can be described by a disjunction over a set of Boolean variables. Since MAXSAT is NP-hard, the mechanism does not have time to find an outcome that satisfies the maximum number of simultaneously satisfiable agents so it must settle for an outcome that is approximately optimal. Section 6 provides a proof that the best a mechanism can guarantee is that half of the maximum number of simultaneously satisfiable agents will be satisfied. This is true for implementation in the four main equilibrium concepts used in the study of mechanism design in complete information environments.

We are not the first to study approximation algorithms in mechanism design. Nisan and Ronen (Nisan and Ronen, 1999) recently studied mechanism design for the task allocation problem under dominant strategy equilibrium. In (Nisan and Ronen, 2000), Nisan and Ronen show that, under certain conditions, dominant strategy implementation of social choice rules that approximately maximize the sum of the agents' utilities is impossible. Our work extends the idea of computational mechanism design to Nash implementation. The negative results of Section 6 combined with Nisan and Ronen's impossibility result indicate that our ability to find good approximation mechanisms for a problem may be limited even when there exist good approximation algorithms for the non-multiagent version of the same problem.

It certainly could be argued that the multiagent MAXSAT problem as we define it is somewhat restrictive since each agent is limited to preferences defined by simple disjunctions rather than more general Boolean formulae. Our goal, however, is not to develop mechanisms for a particular application. Rather, our interest is in determining how restricting the agents and the mechanism to polynomial time computation affects the existing results in the economics literature on mechanism design. In particular, we would like to determine which mechanisms in the literature are polynomial time when the social choice rule consists of approximately optimal solutions to an NP-hard optimization problem. With these goals in mind and in order to focus our analysis, we need a conceptually simple NP-hard problem with many known approximation algorithms. Multiagent MAXSAT satisfies these requirements. On the other hand, we know of no approximation algorithms for the version of this problem in which the agents' preferences are defined by more general Boolean formulae.

2. MECHANISM DESIGN

We begin with a formal description of the mechanism design problem. This section is based on material from (Mas-Colell et al., 1995), (Maskin, 1985), and (Moore, 1992).

Definition 1 *A* **mechanism design problem** *consists of the following:*

- *a set of I* **agents** *that must make a collective choice over some finite set X of possible outcomes.*
- *for each agent i, a set Θ_i of possible* **types**. *An agent's type determines its preferences over the outcomes. It can be thought of as the characteristics of the agent. A vector indicating the type of each agent is called a* **type profile**.
- *for each agent i, a* **utility function** $u_i : X \times \Theta_i \rightarrow \Re$ *that represents the agent's preferences over the outcomes given the agent's type. Each agent is assumed to be trying to maximize its utility.*
- *a* **social choice rule** $F : \Theta_1 \times \ldots \times \Theta_I \rightarrow 2^X \setminus \{\emptyset\}$. *If $F(\cdot)$ is single valued it is called a* **social choice function**.[2] *A social choice rule maps a type profile to a nonempty set of outcomes that are considered socially desirable given that type profile for the agents.*

Definition 2 *A* **mechanism** $\Gamma = (A_1, \ldots, A_I, g(\cdot))$ *consists of a collection of action sets $A_1, \ldots, A_I$ and an outcome function $g : A_1 \times \ldots \times A_I \rightarrow X$. Each agent i chooses an action from set A_i. The mechanism then sets the outcome to $g(a_1, \ldots, a_I)$.*

The goal of the mechanism designer is to ensure that the outcomes of the mechanism are socially desirable as defined by the social choice rule $F(\cdot)$. For example, consider the following multiagent version of the MAXSAT problem:

Definition 3 *The* **Multiagent** MAXSAT *problem is defined as follows:*

- *The parameters of the problem consist of a set of Boolean variables.*
- *The set of outcomes consists of all truth assignments to those variables.*
- *An agent's type consists of preferences over the truth assignments. The preferences are restricted to those that can be described by a simple disjunction over a subset of the variables and their negations. We will refer to such a disjunction as a clause. Each agent*

prefers that its clause be satisfied but does not differentiate between satisfying truth assignments. A type profile, therefore, is a vector of clauses where each clause represents a single agent's type.

- *The goal of the mechanism is to maximize the total number of satisfied clauses. The social choice rule for this problem is defined by:*

  $$\text{MAXSAT}(\theta) \quad = \quad \{t : t \text{ is an outcome that satisfies } m(\theta) \text{ agents}\}$$

  where $m(\theta)$ is the maximum number of simultaneously satisfiable clauses in the type profile θ.

Consider the robot example again and assume that the states of the world can be represented by truth assignments over a set of Boolean variables. For example, suppose there are two blocks B_1 and B_2 and one table. Let x_i = True represent B_i being on the table and let x_i = False represent B_i being on the floor for $i = 1, 2$. A robot's type corresponds to its goal since the robot's goal determines its preferences over the outcomes. Suppose each robot's goal can be represented by a disjunction over the two variables and their negations. Let the mechanism's goal be to satisfy as many of the robots as possible. This is an example of a multiagent MAXSAT problem. Suppose the mechanism requires the robots to declare their goals. In other words, the action sets are just the set of clauses over the Boolean variables. Nothing prevents the robots from lying to the mechanism since the mechanism has no way to verify what the robots' goals really are. Notice, however, that the social choice rule which the mechanism is trying to satisfy is a mapping of the robots' true types not the declared types. The mechanism wants to choose a truth assignment that maximizes the total number of true clauses that are satisfied regardless of the clauses the robots declared. To achieve its goal, the mechanism must be designed in such a way that the robots are enticed to reveal enough information about their true types for the mechanism to make an appropriate decision.

The situation that we study in this paper is one in which the mechanism does not have time to choose an optimal outcome so it must settle for an approximately optimal outcome. For example, suppose the mechanism uses Johnson's first approximation algorithm[3] to determine the outcome. This is a greedy algorithm that takes the literal that appears in the most clauses and sets it to True. It then repeatedly chooses the unassigned literal that appears most in the remaining unsatisfied clauses and sets that to True. Let ties be broken by choosing the least numbered variable first and assigning True before False. If there are five robots with goals defined by the vector $\theta = (x_1, \bar{x}_1 \vee \bar{x}_2, \bar{x}_1 \vee \bar{x}_2, x_2, x_2)$

then the mechanism chooses $t = \bar{x}_1 x_2$. Robot 1 is not satisfied by this outcome. However, if Robot 1 declared its type to be $x_1 \vee \bar{x}_2$, the outcome would be $t = x_1 \bar{x}_2$ which does satisfy Robot 1. Therefore, in this instance, it is better for Robot 1 to lie about his goal.

The literature considers two types of environments with regard to the information that the agents possess about each other. In the *complete information* case, which is the case we consider in this paper, each agent knows his own type as well as the types of all the other agents. In the *incomplete information* case, each agent has no information or only partial information about the other agents' types. In either case, the mechanism does not know the agents' types. A mechanism $\Gamma = (A_1, \ldots, A_I, g(\cdot))$ combined with a set of possible types Θ_i and a utility function u_i for each agent i induces a game of complete or incomplete information depending on the information available to the agents. At the beginning of the game, each agent somehow comes to know its type. Formally, we say each agent receives a signal that indicates its type. Each agent then chooses an action based on its type. In the complete information case, a strategy for agent i is a mapping $s_i : \Theta_1 \times \ldots \times \Theta_I \to A_i$ while in the incomplete information case, a strategy for agent i is a mapping $s_i : \Theta_i \to A_i$.

The mechanism designer's goal is to design the mechanism in such a way that, for every possible type profile, some equilibrium outcome of the induced game is socially desirable. There are many competing equilibrium concepts. In this paper, we will be discussing dominant strategy and Nash equilibrium which are defined below. In these definitions and in the remainder of this paper, we use the notation a_{-i} to indicate the vector $a = (a_1, \ldots, a_I)$ with the i-th component removed. We use (a'_i, a_{-i}) to represent the vector a with the i-th component replaced by a'_i.

Definition 4 *A strategy profile $s^*(\cdot) = (s_1^*(\cdot), \ldots, s_I^*(\cdot))$ is a* **dominant strategy equilibrium** *of a mechanism $\Gamma = (A_1, \ldots, A_I, g(\cdot))$ if, for each agent i and all type profiles $\theta = (\theta_1, \ldots, \theta_I)$,*

$$u_i(g(s_i^*(\theta), a_{-i}), \theta_i) \geq u_i(g(a'_i, a_{-i}), \theta_i)$$

for all $a'_i \in A_i$ and all $a_{-i} \in A_{-i}$.

In other words, a strategy profile is a dominant strategy equilibrium if no matter what the other agents do, no agent has another action that can increase its utility.

Definition 5 *A strategy profile $s^*(\cdot) = (s_1^*(\cdot), \ldots, s_I^*(\cdot))$ is a* **Nash equilibrium** *of a mechanism $\Gamma = (A_1, \ldots, A_I, g(\cdot))$ if, for each agent i*

and all type profiles $\theta = (\theta_1, \ldots, \theta_I)$,

$$u_i(g(s_i^*(\theta), s_{-i}^*(\theta)), \theta_i) \geq u_i(g(a_i', s_{-i}^*(\theta)), \theta_i)$$

for all $a_i' \in A_i$.

In other words, a strategy profile is a Nash equilibrium if no agent can unilaterally deviate from the prescribed action and increase its utility.

Definition 6 *An* **equilibrium outcome** *of a mechanism* Γ *for some type profile* θ *is the outcome produced by an equilibrium strategy profile* $s^*(\theta)$ *of* Γ.

There are different degrees to which a mechanism can satisfy a social choice rule. These are defined as follows:

Definition 7 *Given a mechanism* Γ*, let* $E(\theta)$ *be the set of equilibrium outcomes for a type profile* θ*. We say*

(i) Γ **implements** $F(\cdot)$ *if* $E(\theta) \cap F(\theta) \neq \emptyset$ *for all* θ.

(ii) Γ **strongly implements** $F(\cdot)$ *if* $\emptyset \neq E(\theta) \subset F(\theta)$ *for all* θ.

(iii) Γ **fully implements** $F(\cdot)$ *if* $E(\theta) = F(\theta)$ *for all* θ.

If multiple equilibria exist, it might be difficult to argue that one equilibrium will be played while another will not since all equilibria are equally rational according to any particular equilibrium criterion. The definition of implementation does not exclude the possibility that undesirable equilibrium outcomes are possible. Strong implementation eliminates mechanisms that do not guarantee that every equilibrium outcome is socially desirable. Full implementation receives a great deal of attention in the literature on Nash implementation. However, in line with the philosophy of approximation algorithms, we want to guarantee that all outcomes reach some threshold of acceptability. We are not necessarily concerned with making all acceptable outcomes possible. In designing mechanisms to control the collective decision making process of a group of autonomous agents, strong implementation should suffice.

3. POLYNOMIAL TIME MECHANISMS

From a computational perspective, the mechanism design problem consists of two parts: the computation performed by the mechanism and the computation performed by the agents. From the mechanism's point of view, an instance of the problem consists of the description of the problem's parameters plus a description of the actions chosen by

the agents. From the agent's point of view, an instance of the problem consists of the description of the problem's parameters and a description of the agent's type.[4] Hence, we have the following definitions:

Definition 8 $\Gamma = (A_1, \ldots, A_I, g(\cdot))$ *is a* **polynomial time mechanism** *if there is some polynomial* $p(\cdot)$ *such that* $g(a_1, \ldots, a_I)$ *is computable in time which is* $O(p(|a_1| + \ldots + |a_I| + l))$ *where* $|a_i|$ *is the size of the binary description of action* a_i *and* l *is the size of the binary description of the problem's parameters.*

Definition 9 *A strategy profile* $s(\cdot) = (s_1(\cdot), \ldots, s_I(\cdot))$ *is a* **polynomial time strategy profile** *if there is a polynomial* $p(\cdot)$ *such that for all* i, $s_i(\theta)$ *is computable in time which is* $O(p(|\theta| + l))$ *where* $|\theta|$ *is the size of the binary description of the agents' type profile and* l *is the size of the binary description of the problem's parameters.*

We are interested in situations in which the computation performed by the mechanism and the agents is restricted to polynomial time. This leads to the following variations on Definition 7.

Definition 10 *Given a polynomial time mechanism* Γ, *let* $\mathrm{PE}(\theta)$ *be the set of polynomial time equilibrium outcomes for a type profile* θ. *If* Γ *is a polynomial time mechanism then we say that,* **in polynomial time for polynomial time bounded agents**,

(i) Γ **implements** $F(\cdot)$ *if* $\mathrm{PE}(\theta) \cap F(\theta) \neq \emptyset$ *for all* θ.

(ii) Γ **strongly implements** $F(\cdot)$ *if* $\emptyset \neq \mathrm{PE}(\theta) \subset F(\theta)$ *for all* θ.

(iii) Γ **fully implements** $F(\cdot)$ *if* $\mathrm{PE}(\theta) = F(\theta)$ *for all* θ.

3.1. REVELATION MECHANISMS

Mechanisms in which $A_i = \Theta_i$ for all i form an important class of mechanisms known as *revelation mechanisms*. In other words, revelation mechanisms require the agents to declare their types. We say that a social choice rule $F(\cdot)$ is truthfully implementable if there is a revelation mechanism for which truthful type declarations by all the agents constitutes an equilibrium with an outcome in $F(\cdot)$. More formally:

Definition 11 *A social choice rule* $F(\cdot)$ *is* **truthfully implementable** *if there is a revelation mechanism* $\Gamma = (\Theta_1, \ldots, \Theta_I, g(\cdot))$ *that has an equilibrium strategy profile* $s^*(\cdot)$ *such that* $s_i^*(\theta) = \theta_i$ *and* $g(s^*(\theta)) \in F(\theta)$ *for all type profiles* θ *and all agents* i.

Definition 12 *A revelation mechanism* Γ *is said to be* **truthful** *if truth telling is a dominant strategy for* Γ.

The following result known as the *Revelation Principle* states that truthfully implementable social choice rules are the only social choice rules that can be implemented. This principle applies to many equilibrium concepts including dominant strategy and Nash equilibrium so we state it without specifying a particular equilibrium concept (see the discussion in (Maskin, 1985) on p. 182-183):

Proposition 1 (The Revelation Principle) *Suppose there is a mechanism* Γ *that implements social choice rule* $F(\cdot)$*. Then* $F(\cdot)$ *is truthfully implementable.*

Proof. See for example (Mas-Colell et al., 1995). ∎

The Revelation Principle has a polynomial time analog if the agents are restricted to polynomial time strategies. This result holds for any equilibrium concept for which the standard Revelation Principle applies.

Proposition 2 (The Polynomial Time Revelation Principle) *If a social choice rule* $F(\cdot)$ *is implementable in polynomial time for polynomial time bounded agents then* $F(\cdot)$ *is truthfully implementable in polynomial time for polynomial time bounded agents.*

Proof. See (O'Connell and Stearns, 2000). ∎

Using the Polynomial Time Revelation Principle, we can show that MAXSAT$(\cdot)$, defined in Definition 3, is not polynomial time implementable.

Proposition 3 MAXSAT$(\cdot)$ *is not polynomial time implementable for polynomial time bounded agents assuming* P $\neq$ NP*.*

Proof. See (O'Connell and Stearns, 2000). ∎

Since MAXSAT$(\cdot)$ is not implementable in polynomial time for polynomial time bounded agents, the best one can hope for is to find some approximation to MAXSAT$(\cdot)$ that is implementable in polynomial time for polynomial time bounded agents. An approximation to MAXSAT$(\cdot)$ is defined as a social choice rule of the form:

$$c\text{-MAXSAT}(\theta) = \{t : t \text{ satisfies at least } cm(\theta) \text{ clauses}\}$$

where $c \in (0, 1)$ and $m(\theta)$ is the maximum number of simultaneously satisfiable clauses in θ. In general, if $F(\cdot)$ is a social choice rule taking the form

$$F(\theta) = \{t \in X : t \in \arg\max_{x \in X} h(x, \theta)\}$$

for some real valued function $h(\cdot)$, a c-approximation for $F(\cdot)$ is a social choice rule c-$F(\theta)$ such that

$$c\text{-}F(\theta) = \{t \in X : h(t, \theta) \geq c \max_{x \in X} h(x, \theta)\}$$

where c is a constant between 0 and 1.

Definition 13 *A social choice rule $F(\cdot)$ is* **approximately implementable** *if there is some constant $c \in (0, 1)$ such that c-$F(\cdot)$ is implementable.*

In Sections 4 and 5, we examine whether there exists a constant $c \in (0, 1)$ such that c-MAXSAT$(\cdot)$ is implementable in dominant strategy or Nash equilibrium.

4. DOMINANT STRATEGY IMPLEMENTATION

When dealing with dominant strategy implementation, one must contend with an impossibility theorem known as the Gibbard-Satterthwaite Theorem (Gibbard, 1973; Satterthwaite, 1975) which restricts the set of implementable social choice functions to those that are dictatorial.

Definition 14 *A social choice function $f(\cdot)$ is* **dictatorial** *if there is a single agent i such that, for all type profiles θ, $f(\theta)$ is agent i's most preferred outcome.*

Proposition 4 (The Gibbard-Satterthwaite Theorem) *Let $f(\cdot)$ be any social choice function. Suppose the set of possible outcomes, X, is finite and contains at least three elements and that the range of $f(\cdot)$ is X. Further suppose that the set of possible preference relations over X contains the set of strict preferences over X. Then $f(\cdot)$ is truthfully implementable in dominant strategies if and only if $f(\cdot)$ is dictatorial.*

Since the set of preference relations for MAXSAT does not include the set of strict preference relations, the Gibbard-Satterthwaite Theorem does not apply to multiagent MAXSAT. To see this, observe that if an agent's clause does not include a variable x_i then the agent is indifferent between truth assignments that are identical except in their assignment to x_i. Furthermore, if the agent's clause includes more than one literal then the agent is indifferent between truth assignments that satisfy at least one of the literals. As the following proposition shows, MAXSAT$(\cdot)$ is truthfully implementable in dominant strategies.

Proposition 5 *There is a revelation mechanism with a non-dictatorial outcome function that truthfully implements* MAXSAT(·) *in dominant strategies.*

Proof. See (O'Connell and Stearns, 2000). ∎

We know from Proposition 3 that MAXSAT(·) cannot be implemented in polynomial time so we are interested in determining whether there is some constant c such that c-MAXSAT(·) can be implemented in polynomial time. The question is how to convert one of the many existing approximation algorithms for MAXSAT into a mechanism that truthfully implements c-MAXSAT(·). In the remainder of this section, we show that we can strongly implement $\frac{1}{2}$-MAXSAT(·) in polynomial time using an algorithm we call the complement algorithm.

The following lemma implies that there is a simple 1/2-approximation algorithm for MAXSAT.

Lemma 1 *For a truth assignment t, let $\bar{t}$ denote the truth assignment such that for every variable v, $\bar{t}(v) =$ True if and only if $t(v) =$ False. For all truth assignments t, either t or $\bar{t}$ satisfies at least half the maximum number of satisfiable clauses.*

Proof. Let θ_i be any clause that t does not satisfy. Let l_i be a literal in θ_i. Then $t(l_i) =$ False which implies $\bar{t}(l_i) =$ True. Therefore, $\bar{t}$ satisfies θ_i. ∎

We can use this property to develop a polynomial time mechanism that strongly implements $\frac{1}{2}$-MAXSAT(·). Fix a truth assignment t and define a social choice function $f(\cdot)$ as follows:

$$f(\theta) \quad = \quad \begin{cases} \bar{t} & \text{if } \bar{t} \text{ satisfies more clauses in } \theta \text{ than } t \\ t & \text{otherwise} \end{cases} \tag{1}$$

This social choice function is not dictatorial since for each agent i we can define θ such that $\bar{t}$ does not satisfy θ_i but does satisfy every clause in θ_{-i} while t does not satisfy any clause in θ_{-i}. For example, let $t = x_1\bar{x}_2\bar{x}_3$ and $\theta = (x_1, x_2, x_3)$. We have $f(\theta) = \bar{t}$ which is not agent 1's most preferred outcome. A similar argument applies to the other two agents which implies that $f(\theta)$ is not the same agent's most preferred outcome for every θ. Lemma 1 implies that $f(\theta)$ is a $\frac{1}{2}$-approximation for MAXSAT(·). Therefore, any mechanism that truthfully implements $f(\cdot)$, truthfully implements $\frac{1}{2}$-MAXSAT(·).

Define the following revelation mechanism:

Mechanism 1

1. *Given truth assignment t from the definition of $f(\cdot)$ in Equation 1.*

2. If t satisfies at least as many of the declared clauses as $\bar{t}$
3. then choose t as the outcome
4. else choose $\bar{t}$.

Theorem 1 *Mechanism 1 above truthfully implements $f(\cdot)$ in dominant strategies in polynomial time for polynomial time bounded agents.*

Proof. Mechanism 1 is polynomial time since, to compute the outcome, it need only compare the number of clauses in θ that are satisfied by two fixed truth assignments. Truth telling is certainly a polynomial time strategy so it suffices to show that truth telling constitutes a dominant strategy equilibrium.

Let t be the truth assignment used in the definition of $f(\cdot)$ in Equation 1. Let θ_i be agent i's true type.

Case 1: t satisfies θ_i and $\bar{t}$ doesn't.

Removing literals from θ_i potentially decreases the number of clauses satisfied by t and does not affect the number of clauses satisfied by $\bar{t}$. Adding literals cannot increase the number of clauses satisfied by t and cannot decrease the number of clauses satisfied by $\bar{t}$. Since lying either leaves the number of clauses satisfied by t the same or decreases it and leaves the number of clauses satisfied by $\bar{t}$ the same or increases it, lying either leaves the outcome unaffected or results in an outcome that is worse for agent i.

Case 2: $\bar{t}$ satisfies θ_i and t doesn't. A symmetric argument applies to this case.

Case 3: Both t and $\bar{t}$ satisfy θ_i. In this case, agent i does not care which of the two truth assignments is chosen so lying cannot be beneficial or harmful.

Since lying is never beneficial, truth telling is a dominant strategy. ∎

Corollary 1 *The social choice rule $\frac{1}{2}$-MAXSAT$(\cdot)$ is strongly implementable in dominant strategies in polynomial time for polynomial time bounded agents.*

Proof. Case 3 is the only case in the proof of Theorem 1 in which lying can affect the outcome and not be harmful to the agent. In (O'Connell and Stearns, 2000), we show that any lie in Case 3 results in an outcome that is in $\frac{1}{2}$-MAXSAT(θ). ∎

5. NASH IMPLEMENTATION

We now consider mechanism design in environments with complete information using Nash equilibrium. There are two properties of social choice rules that are extremely important for Nash implementation – no veto power and monotonicity. No veto power states that one agent

cannot override the wishes of all the other agents. Monotonicity says that whenever the type profile changes from θ to θ' and the set of outcomes that an alternative $t \in F(\theta)$ is preferred to remains the same or expands then t must also be in $F(\theta')$. These two properties are formalized in the following definitions.

Definition 15 *A social choice rule $F(\cdot)$ satisfies* **no veto power** *if for any type profile θ such that $I-1$ agents rank an alternative t as their weakly most preferred choice, then $t \in F(\theta)$.*

Let $L_i(t, \theta_i) = \{t' : u_i(t, \theta_i) \geq u_i(t', \theta_i)\}$. $L_i(t, \theta_i)$ is called agent i's *lower contour set* for t.

Definition 16 *A social choice rule $F(\cdot)$ is* **monotonic** *if for all type profiles θ and θ' and all outcomes t such that $t \in F(\theta)$ and $L_i(t, \theta_i) \subseteq L_i(t, \theta_i')$ for all i, we have $t \in F(\theta')$.*

The following result is known as Maskin's Theorem (see (Maskin, 1985)).

Proposition 6 (Maskin's Theorem) *If a social choice rule is fully Nash implementable then it is monotonic. If there are at least three agents then a social choice rule that is monotonic and satisfies no veto power is fully Nash implementable.*

Monotonicity is also required for full Nash implementation in polynomial time for polynomial time bounded agents.

Proposition 7 *If a social choice rule $F(\cdot)$ is fully Nash implementable in polynomial time for polynomial time bounded agents then it is monotonic.*

Proof. See (O'Connell and Stearns, 2000). ∎

Proposition 8 *If there are at least two variables then* MAXSAT$(\cdot)$ *is not monotonic.*

Proof. We provide a proof for the case of two agents and two variables. It can be generalized easily. Let $\theta = (x_1, \bar{x}_1)$. The maximum number of simultaneously satisfiable clauses in θ is 1. Let $t = x_1 x_2$ which is in MAXSAT(θ). Let $\theta' = (x_2, \bar{x}_1)$. Since t satisfies agent 1 when the type profile is either θ or θ', $L_1(t, \theta_1) = L_1(t, \theta_1')$. Since agent 2's type is the same in either case, $L_2(t, \theta_2) = L_2(t, \theta_2')$. However, $t \notin$ MAXSAT(θ') which implies that MAXSAT$(\cdot)$ is not monotonic. ∎

Even if MAXSAT$(\cdot)$ were monotonic, by Proposition 3, it cannot be implemented in polynomial time for polynomial time bounded agents

assuming P $\neq$ NP. If the mechanism and the agents are restricted to polynomial time computation, the best one could hope for is to approximately implement MAXSAT($\cdot$). Therefore, we need to find a constant c such that c-MAXSAT($\cdot$) is monotonic even though MAXSAT($\cdot$) is not. We also need to find a polynomial time mechanism that implements c-MAXSAT($\cdot$).

There are several different mechanisms used to prove Maskin's theorem (Maskin, 1985; Repullo, 1987; Saijo, 1988; McKelvey, 1989). For example, in (Saijo, 1988), Saijo defines a mechanism that fully implements a monotonic social choice rule satisfying no veto power as follows. The action sets are defined by $A_i = \Theta_i \times \Theta_{i+1} \times X \times \{1, \ldots, I\}$. Each agent i declares his own type θ_i^i, the type of his neighbor θ_i^{i+1} where we take $I + 1 = 1$, an outcome x_i, and a number k_i between 1 and I. The outcome function $g(\cdot)$ is defined by the following rules:

Mechanism 2 (Saijo's Mechanism)

Let $a = \left[(\theta_i^i, \theta_i^{i+1}, x_i, k_i)\right]_{i=1}^{I}$ *be the action profile.*

Rule I: If $\theta_i^i = \theta_{i-1}^i$ *and* $x_i = x$ *for all* i *and* $x \in F(\theta_1^1, \ldots, \theta_I^I)$
then $g(a) = x$.

Rule II: If $\theta_i^i = \theta_{i-1}^i$ *for all* i *except* j *or* $j+1$,
and $x_i = x$ *for all* i *except* j,
and $x \in F(\theta_1^1, \ldots, \theta_{j-1}^{j-1}, \theta_{j-1}^j, \theta_{j+1}^{j+1}, \ldots, \theta_I^I)$

$$\text{then } g(a) = \begin{cases} x_j & \text{if } x_j \in L_j(x, \theta_{j-1}^j), \\ x & \text{otherwise} \end{cases}$$

Rule III: If neither Rule I nor Rule II applies
then set $g(a) = x_n$ *where* $n = \left(\sum_{i \in I} k_i\right) (\mathrm{mod} I) + 1$.

Notice that Mechanism 2 checks whether a given outcome is socially desirable given the declared type profile. This check, which also appears in the mechanisms used in (Repullo, 1987) and (Maskin, 1985) corresponds to the following decision problem when $F(\cdot)$ is c-MAXSAT($\cdot$).

Definition 17 ***c*-MAXSAT Membership Problem**
Given a set of Boolean variables $V = \{v_1, \ldots, v_n\}$, *a set of clauses* θ *over* V *and a truth assignment* t, *does* t *satisfy at least* c *times the maximum number of satisfiable clauses in* θ*?*

Proposition 9 shows that this problem is NP-hard which implies that Saijo's mechanism with $F(\cdot) = c$-MAXSAT($\cdot$) is not a polynomial time mechanism unless P = NP.

Proposition 9 *Let* $c \in (0, 1)$ *be such that there is a polynomial time approximation algorithm for c-MAXSAT. The c-MAXSAT Membership problem is NP-hard.*

Proof. See (O'Connell and Stearns, 2000). ∎

Theorem 2 *Let $c \in (0,1)$ be such that there is a polynomial time approximation algorithm for c-MAXSAT. When $F(\cdot) = c\text{-MAXSAT}(\cdot)$, Saijo's mechanism is not a polynomial time mechanism unless* P = NP.

In Saijo's mechanism, for a strategy profile to be an equilibrium strategy profile, it must always be the case that all but at most one of the agents choose an outcome that is in $F(\theta)$. Theorem 2 implies that if c is such that Saijo's mechanism is polynomial time then the agents cannot find an alternative in c-MAXSAT$(\cdot)$ in polynomial time. Therefore, either the mechanism is not polynomial time or there is no polynomial time equilibrium strategy profile. Whether there is a general mechanism to implement any monotonic social choice rule satisfying no veto power that does not check membership in $F(\cdot)$ is an open question.

As discussed in Section 2, we are willing to settle for strong implementation as opposed to full implementation of c-MAXSAT$(\cdot)$. Thus, it is sufficient for our purposes to find a social choice rule $F(\cdot)$ such that

1. $F(\cdot)$ is monotonic
2. $F(\cdot)$ satisfies no veto power
3. $F(\theta) \subseteq c\text{-MAXSAT}(\theta)$ for all θ
4. checking membership in $F(\cdot)$ is easy.

Such a social choice rule exists for $c = 1/2$.

Theorem 3 *Let $F(\theta)$ be the set of outcomes t such that t satisfies at least $\frac{I}{2}$ clauses in θ. If there are at least three agents, $F(\cdot)$ is fully Nash implementable using Saijo's mechanism.*

Proof. $F(\cdot)$ clearly satisfies no veto power since there are at least three agents. Let θ, θ' and t be such that $t \in F(\theta)$ and $L_i(t, \theta_i) \subseteq L_i(t, \theta'_i)$ for all i. Since every clause has a satisfying truth assignment, if t satisfies θ_i then t must satisfy θ'_i. Therefore, t satisfies at least half of the clauses in θ' so $t \in F(\theta')$. Hence, $F(\cdot)$ is monotonic. Since $F(\cdot)$ is monotonic and satisfies no veto power, Saijo's mechanism fully Nash implements $F(\cdot)$. ∎

Corollary 2 *The social choice rule $\frac{1}{2}$-MAXSAT$(\cdot)$ is strongly Nash implementable in polynomial time for polynomial time bounded agents.*

Proof. Let $F(\cdot)$ be the the social choice rule used in Theorem 3. It is easy to check whether a truth assignment t is in $F(\theta)$ so Saijo's mechanism is polynomial time computable. Let θ be the agents' true

type profile. For any truth assignment t such that $t \in F(\theta)$ and any $k, 1 \leq k \leq I$, let $s_i(\theta) = (\theta_i, \theta_{i+1}, t, k)$ for all i. The strategy profile $s(\theta) = (s_1(\theta), \ldots, s_I(\theta))$ is a Nash equilibrium.[5] Such a strategy profile is easy for the agents to compute since the agents can use the complement algorithm to find a $t \in F(\theta)$. Therefore, $F(\cdot)$ is strongly Nash implementable in polynomial time for polynomial time bounded agents. Since $F(\theta)$ is a subset of $\frac{1}{2}$-MAXSAT(θ) for all θ, $\frac{1}{2}$-MAXSAT$(\cdot)$ is strongly Nash implementable in polynomial time for polynomial time bounded agents. ∎

6. UPPER BOUNDS ON APPROXIMABILITY

Existing work in mechanism design shows that the set of social choice rules that are implementable in Nash equilibrium is much smaller than the set of social choice rules that are implementable in refinements such as undominated Nash equilibrium[6] (Palfrey and Srivastava, 1991; Jackson et al., 1994) and subgame perfect equilibrium[7] (Moore and Repullo, 1988). In (Palfrey and Srivastava, 1991), it is observed that for a social choice rule to be fully implementable in Nash, undominated Nash, or subgame perfect equilibrium, it must satisfy Property Q defined below. Property Q must also be satisfied for full implementation in dominant strategies. In (O'Connell and Stearns, 2000), we show that Property Q must be satisfied for full implementation in polynomial time for polynomial time bounded agents.

Definition 18 *(Palfrey and Srivastava, 1991) A social choice rule satisfies* **Property Q** *if, whenever θ, θ' and t are such that $t \in F(\theta)$ and $t \notin F(\theta')$, there is an i such that $\theta_i \neq \theta'_i$ and agent i is not completely indifferent under θ'_i.*

Theorem 4 *Let $F(\cdot)$ be a social choice rule such that, for some constant $c \in (0,1)$, $F(\theta) \subseteq c$-MAXSAT(θ) for all θ. Suppose for some type profile θ there exists a truth assignment $t \in F(\theta)$ that satisfies less than cI clauses. Then $F(\cdot)$ does not satisfy Property Q.*

Proof. Let $F(\cdot)$, θ, c, and t be as defined above. For each i such that t satisfies θ_i, let l_i be any literal in θ_i such that $t(l_i) =$ True. Create θ' from θ by adding $\bar{l}_i$ to θ_i for each clause θ_i that t satisfies. For each i, either $\theta'_i = \theta_i$ or agent i is completely indifferent under type θ'_i. Hence, for $F(\cdot)$ to satisfy Property Q, it must be the case that $t \in F(\theta')$.

Let $m(\theta')$ be the maximum number of clauses that can be satisfiable simultaneously in the type profile θ'. Since $\bar{t}$ must satisfy every clause that t does not satisfy, and $\bar{t}$ must satisfy θ'_i if t satisfies θ_i, $m(\theta') =$

I. Since the clauses in θ that t doesn't satisfy also appear in θ', the number of clauses in θ' that t satisfies is strictly less than $cI = cm(\theta')$. Therefore, $t \notin c\text{-MAXSAT}(\theta')$. This implies that $t \notin F(\theta')$ since $F(\theta) \subseteq c\text{-MAXSAT}(\theta)$ for all θ. Thus, $F(\cdot)$ does not satisfy Property Q. ∎

Theorem 4 says that the only approximate social choice rules for MAXSAT that can be fully implemented are those that guarantee the number of clauses that the outcomes satisfy is a constant times the total number of clauses rather than a constant times the maximum number of simultaneously satisfiable clauses. The following corollary shows that this is impossible for $c > 1/2$:

Corollary 3 *For any c such that $\lceil cI \rceil > \lceil \frac{I}{2} \rceil$, c-MAXSAT$(\cdot)$ cannot be strongly implemented in dominant strategy, Nash, undominated Nash or subgame perfect equilibrium.*

Proof. Let Γ be a mechanism that strongly implements c-MAXSAT$(\cdot)$ for some constant c such that $\lceil cI \rceil > \lceil \frac{I}{2} \rceil$. Define $E : \Theta_1 \times \ldots \times \Theta_I \rightarrow X$ such that $E(\theta)$ is the set of equilibrium outcomes of Γ for a type profile θ. By definition, Γ fully implements $E(\cdot)$. Since Γ strongly implements c-MAXSAT$(\cdot)$, $E(\theta) \subseteq c\text{-MAXSAT}(\theta)$ for all θ.

Let $\theta'_i = x_1$ for $1 \leq i \leq \lfloor \frac{I}{2} \rfloor$. Let $\theta'_i = \bar{x}_1$ for $\lfloor \frac{I}{2} \rfloor + 1 \leq i \leq I$. Since the maximum number of simultaneously satisfiable clauses is $\lceil \frac{I}{2} \rceil$, no truth assignment can satisfy more than $\lceil \frac{I}{2} \rceil$ clauses. Therefore, any $t \in E(\theta')$ satisfies less than cI clauses in θ'. According to Theorem 4, this implies that $E(\cdot)$ does not satisfy Property Q. But then $E(\cdot)$ is not fully implementable which is a contradiction. ∎

7. CONCLUSION

We have formalized the computational problem of mechanism design in such a way that classic results from the field can be applied. Using a multiagent version of MAXSAT, we have investigated the difficulties that arise in applying these results to NP-hard optimization problems. We have demonstrated that, despite the impossibility results regarding dominant strategy implementation, it is possible to implement an approximate social choice rule for MAXSAT in dominant strategy equilibrium.

With regard to Nash implementation, we showed that approximation was beneficial in two ways. It enabled strong Nash implementation when the exact social choice rule was not monotonic and it enabled polynomial time implementation when finding a member of the exact rule was NP-hard. In addition, we showed that the standard mechanisms[8] used to prove Maskin's theorem are not polynomial time when the social

choice rule is c-MAXSAT($\cdot$) for any c such that there is a polynomial time c-approximation algorithm for MAXSAT. This implies that these mechanisms do not fully Nash implement c-MAXSAT($\cdot$) for any c in polynomial time for polynomial time bounded agents. We also showed that the best approximate social choice rule for MAXSAT($\cdot$) that can be strongly implemented in dominant strategy, Nash, undominated Nash or subgame perfect equilibrium is $\frac{1}{2}$-MAXSAT($\cdot$). This is somewhat remarkable given that there are many approximation algorithms for the non-multiagent version of this problem that achieve better lower bounds than 1/2. Several authors (Yannakakis, 1994; Goemans and Williamson, 1995; Asano, 1997) provide algorithms for MAXSAT that achieve lower bounds $\geq \frac{3}{4}$. Our results, therefore, indicate that it can be much more difficult to design good approximation mechanisms than to design good approximation algorithms. It may be the case that these results are peculiar to MAXSAT so future work should consider multiagent versions of other NP-hard problems. However, we have defined multiagent MAXSAT somewhat restrictively with each agent limited to preferences defined by unweighted disjunctions. The fact that such difficulties arise even for this version of the problem suggests that we will be much less successful using approximation to overcome hardness results in settings with self-interested agents than in traditional computational settings.

Acknowledgments

We would like to thank Professor Laurence Kranich of the Department of Economics at the University at Albany for many helpful comments and suggestions.

Notes

1. This example is taken from (Rosenschein and Zlotkin, 1994).

2. We use lower case $f(\cdot)$ when discussing social choice functions and upper case $F(\cdot)$ when discussing social choice rules.

3. See (Johnson, 1974) or (Battiti, 1998).

4. It should also include a description of the mechanism but we will assume that either the description of the mechanism is fixed or that its size is polynomial in the size of the problem's parameters.

5. See (Saijo, 1988) p. 698.

6. An undominated Nash equilibrium is a Nash equilibrium in which no agent plays a weakly dominated strategy.

7. Subgame perfect equilibrium is defined for games that are played in stages. A strategy profile is a subgame perfect equilibrium if it is a Nash equilibrium in every subgame.

8. We are currently evaluating the complexity of the mechanism due to McKelvey (McKelvey, 1989).

References

Asano, T. (1997). Approximation algorithms for MAX SAT: Yannakakis vs. Goemans and Williamson. In *Proceedings of the 3rd Israel Symposium on Theory and Computing Systems*, pages 24–37, Ramat Gan, Israel.

Battiti, R. (1998). Approximation algorithms and heuristics for MAX-SAT. In Zhu, D.-Z. and Pardalos, P., editors, *Handbook of Combinatorial Optimization*, volume 1, pages 77–148. Kluwer Academic Publishers, Norwell, MA.

Gibbard, A. (1973). Manipulation of voting schemes: A general result. *Econometrica*, 41:587–602.

Goemans, M. and Williamson, D. (1995). Improved approximation algorithms for maximum cut and satisfiabilty problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145.

Jackson, M., Palfrey, T., and Srivastava, S. (1994). Undominated Nash implementation in bounded mechanisms. *Games and Economic Behavior*, 6:474–501.

Johnson, D. (1974). Approximation algorithms for combinatorial problems. *Journal of Computer and Systems Sciences*, 9:256–278.

Mas-Colell, A., Whinston, M., and Green, J. (1995). *Microeconomic Theory*. Oxford University Press, New York, NY.

Maskin, E. (1985). The theory of implementation in Nash equilibrium: a survey. In Hurwicz, L., Schmeidler, D., and Sonnenschein, H., editors, *Social goals and social organization: Essays in memory of Elisha Pazner*, pages 173–204. Cambridge University Press, New York, NY.

McKelvey, R. (1989). Game forms for Nash implementation of general social choice correspondences. *Social Choice and Welfare*, 6:139–156.

Moore, J. (1992). Implementation, contracts and renegotiation in environments with complete information. In Laffont, J.-J., editor, *Advances in economic theory: Sixth World Congress*, pages 182–282. Cambridge University Press, Cambridge, UK.

Moore, J. and Repullo, R. (1988). Subgame perfect implementation. *Econometrica*, 56:1191–1220.

Nisan, N. and Ronen, A. (1999). Algorithmic mechanism design. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 129–140, Atlanta, GA.

Nisan, N. and Ronen, A. (2000). Computationally feasible VCG mechanisms. Working Paper, Institute of Computer Science, Hebrew University of Jerusalem.

O'Connell, T. and Stearns, R. (2000). Polynomial time mechanism design. Technical Report SUNYA-CS-TR-00-1, Department of Computer Science, University at Albany, SUNY.

Palfrey, T. and Srivastava, S. (1991). Nash implementation using undominated strategies. *Econometrica*, 59:479–501.

Repullo, R. (1987). A simple proof of Maskin's theorem on Nash implementation. *Social Choice and Welfare*, 4:39–41.

Rosenschein, J. and Zlotkin, G. (1994). *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. MIT Press, Cambridge, MA.

Saijo, T. (1988). Strategy space reductions in Maskin's theorem: Sufficient conditions for Nash implementation. *Econometrica*, 56(3):693–700.

Sandholm, T. (1999). Distributed rational decision making. In Weiss, G., editor, *Mutiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, pages 201–258. MIT Press, Cambridge, MA.

Satterthwaite, M. (1975). Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10:187–217.

Yannakakis, M. (1994). On the approximation of maximum satisfiability. *Journal of Algorithms*, 17:475–502.