Voting in Cooperative Information Agent Scenarios: Use and Abuse

Jeffrey S. Rosenschein and Ariel D. Procaccia

School of Engineering and Computer Science
Hebrew University, Jerusalem, Israel
{jeff, arielpro}@cs.huji.ac.il

Abstract. Social choice theory can serve as an appropriate foundation upon which to build cooperative information agent applications. There is a rich literature on the subject of voting, with important theoretical results, and builders of automated agents can benefit from this work as they engineer systems that reach group consensus.

This paper considers the application of various voting techniques, and examines nuances in their use. In particular, we consider the issue of preference extraction in these systems, with an emphasis on the complexity of manipulating group outcomes. We show that a family of important voting protocols is susceptible to manipulation by coalitions in the average case, when the number of candidates is constant (even though their worst-case manipulations are $\mathcal{NP}$-hard).

1 Introduction

Research on the theory of social choice, and in particular on its computational aspects, has become an important pursuit within computer science (CS). Motivating this work is the belief that social choice theory can have direct implications on the building of systems comprised of multiple automated agents. This paper describes some of that research, so it is a paper about voting and manipulation, but it is also inter alia about how computer science can help scientists and mathematicians see questions in new ways, spurring progress in new theoretical and applied directions.

Computer science occupies a unique position with respect to other fields of scientific endeavor. Its idiosyncratic nature should be celebrated and strengthened; it helps to make computer science in general, and its subfield multiagent systems (MAS), among the most exciting of scientific research areas today.

Computer Science is, at one and the same time:

1. An independent field with its own set of fundamental questions (both theoretical and applied). This distinctive blurring of theoretical and applied research, in both computer science and MAS, can be a great strength. There exist established fields of Applied Physics and Applied Mathematics, but there is no Applied Computer Science. Fundamental research certainly exists in those parts of CS closest to mathematics, but it is hard to conceive of

M. Klusch, M. Rovatsos, and T. Payne (Eds.): CIA 2006, LNAI 4149, pp. 33–50, 2006.

a computer scientist, even one who occupies its most mathematical corners, uttering anything analogous to the quote attributed to the mathematician Leonard Eugene Dickson, "Thank God that number theory is unsullied by any application".[1] It's the *aspiration* to be purely theoretical that is absent in computer science.

2. A contributor to other fields of both technological and conceptual enablers; as the much-quoted New York Times article provocatively put it, "All Science is Computer Science" [18]. However, that article was referring mainly to the use of powerful computing in other fields, which is certainly not the core of computer science. One could argue that the more important influence of CS has been in changing how scientists in other fields *view* their problems: computation becomes a basic conceptual model, part of the intellectual toolset of other fields. There have been a number of highly visible examples of this trend, such as in cognitive psychology (e.g., information processing models of memory and attention [11]), attempts to develop computational models of the cell [27], and computational statistical mechanics [17]. The type of work described in this paper has had an influence on political science and sociology.
3. An avid consumer of the results produced by other fields. The interdisciplinary nature of computer science is nowhere more evident than in the area of artificial intelligence (AI), and perhaps nowhere more evident in artificial intelligence than in its subarea of multiagent systems. MAS researchers for the last 20 years have eagerly derived inspiration from fields as diverse as biology, physics, sociology, economics, organization theory, and mathematics. This is appropriate (even necessary), and the field takes a justifiable pride in its interdisciplinary openness.

1.1 Game Theory and Economics in MAS

One of the most exciting trends in computer science (and specifically in MAS) has been the investigation of game theory and economics as tools for automated systems.[2] Beginning with work in the mid-1980s, researchers have turned out a steady drumbeat of results, considering the computational aspects of game theory and economics, and how these fields can be put to appropriate use in the building of automated agents [23,9,29,24,19,15].

In this paper, we explore the use of preference aggregation in multiagent systems. Preference aggregation has deep roots in economics, but what distinguishes the CS work on this issue is the concern for computational issues: how are results arrived at (e.g., equilibrium points)? What is the complexity of the process? Can complexity be used to guard against unwanted phenomena? Does complexity of computation prevent realistic implementation of a technique?

The criteria to be used in evaluating this work (and exploiting its results) needs to take into account the ultimately applied nature of the endeavor. The

[1] Of course, number theory eventually provided the basis for the cryptography embedded throughout the internet, so it was not quite as unsullied as all that (eventually).

[2] We use the terms loosely, to encompass related fields and subfields such as decision theory, mechanism design, and general equilibrium theory.

idealized models of classic game theory might fall short when used normatively or descriptively with regard to human behavior, but for the most part that might remain a theoretical concern. *Application* of these models to automated agents quite urgently argued for their adjustment. To take one example, if computing an equilibrium point in a particular interaction is computationally infeasible, what would be the *meaning* of telling an agent to choose one?

Much of what follows first appeared in [21,22,20], and we use that material (particularly from the first of those papers) freely. We present it as a specific example of how preference aggregation can be handled in automated systems, and how computational issues come to the forefront.

1.2 The Theory and Practice of Preference Aggregation

In multiagent environments, it may be the case that different agents have diverse preferences, and it is therefore important to find a way to aggregate agent preferences. Even in situations where the agents are cooperative, there may still be independent motivations, goals, or perspectives that require them to come to a consensus.

A general scheme for preference aggregation is *voting*[3]: the agents reveal their preferences by ranking a set of candidates, and a winner is determined according to a voting protocol. The candidates can be entities of almost any conceivable sort.

For instance, Ghosh et al. [12] designed an automated movie recommendation system, in which the conflicting preferences a user may have about movies were represented as agents, and movies to be suggested were selected according to a voting scheme (in this example there are multiple winners, as several movies are recommended to the user). The candidates in a virtual election could also be items such as beliefs, joint plans [10], or schedules [14]. In fact, to see the generality of the (automated) voting scenario, consider modern web searching. One of the most massive preference aggregation schemes in existence is Google's PageRank algorithm, which can be viewed as a vote among indexed web pages on candidates determined by a user-input search string; winners are ranked (Tennenholtz [26] considers the axiomatic foundations of ranking systems such as this).

Things are made more complicated by the fact that in many automated settings (as in non-virtual environments) the agents are self-interested, or at the very least bring different knowledge/perspectives to the interaction. Such an agent may reveal its preferences untruthfully, if it believes this would make the final outcome of the elections more favorable for it. In fact, well-meaning agents may even lie in an attempt to improve social welfare [20].[4] In any case, whether the agents are acting out of noble or ignoble motives, there may be an undesirable social outcome. Strategic behavior in voting has been of particular interest to AI researchers [3,5,8,21]. This problem is provably acute: it is known [13,25] that, for elections with three or more candidates, in any voting protocol that

[3] We use the term in its intuitive sense here, but in the social choice literature, "preference aggregation" and "voting" are basically synonymous.

[4] Though if several do it in parallel there may be unintended negative consequences.

is non-dictatorial,[5] there are elections where an agent is better off by voting untruthfully.

Of course, in some systems, particularly centrally-designed systems, strategic behavior can be effectively banned by fiat. We would argue this covers a distinct minority of multiagent systems.

1.3 Nuances

Different voting protocols will often have different outcomes, as well as different properties. Some protocols may be hard to manipulate; others may skew the preferences of voters in particular ways. Many of these phenomena are well-known in social choice theory, such as the effects of run-off voting on who wins an election (a good heuristic: bring your candidate up for a vote as late as possible in the process). Consider as another example one that comes from a paper in this collection [20]. There we discuss the concepts of *distortion* and *misrepresentation*, two (related, but distinct) measures of how well (or badly) a given candidate represents the desires of a voter.

Quoting from [20]:

> [T]he misrepresentation of a social choice function... can be easily reformulated as distortion. In fact, similar results can be obtained, but the latter formulation favors candidates that are ranked last by few voters, whereas the former formulation rewards candidates that are placed first by many voters.

So depending on whether distortion or misrepresentation is used, we may be developing a technique that prefers candidates with many first place votes over one that prefers candidates with few last place votes. The choice is in the hands of the designer, and one or the other may be more natural for some specific domain. Continuing from [20]:

> Consider the meeting scheduling problem discussed in [14]: scheduling agents schedule meetings on behalf of their associated users, based on given user preferences; a winning schedule is decided in an election. Say three possible schedules are being voted on. These schedules, being fair, conflict with at most two of the requirements specified by any user... In this case, having no conflicts at all is vastly superior to having at least one conflict, as even one conflict may prevent a user from attending a meeting. As noted above, this issue is taken into account in the calculation of misrepresentation — emphasis is placed on candidates that were often ranked first.

This type of consideration runs throughout our choices of preference aggregation techniques. The system put into place may have major ramifications on the outcomes. In addition (and this is the main concern of the rest of the paper),

[5] In a dictatorial protocol, there is an agent that dictates the outcome regardless of the others' choices.

the resistance of the system to strategic behavior may influence whether agents truthfully reveal their preferences, and whether a socially desirable candidate is elected.

1.4 Complexity to the Rescue

Fortunately, it is reasonable to make the assumption that automated agents are computationally bounded. Therefore, although in principle an agent may be able to manipulate an election, the computation required may be infeasible. This has motivated researchers to study the computational complexity of manipulating voting protocols. It has long been known [2] that there are voting protocols that are $\mathcal{NP}$-hard to manipulate by a single voter. Recent results by Conitzer and Sandholm [6,4] show that some manipulations of common voting protocols are $\mathcal{NP}$-hard, even for a small number of candidates. Moreover, in [7], it is shown that adding a pre-round to some voting protocols can make manipulations hard (even $\mathcal{PSPACE}$-hard in some cases). Elkind and Lipmaa [8] show that the notion of pre-round, together with one-way functions, can be used to construct protocols that are hard to manipulate even by a large minority fraction of the voters.

In computer science, the notion of hardness is usually considered in the sense of worst-case complexity. Not surprisingly, most results on the complexity of manipulation use $\mathcal{NP}$-hardness as the complexity measure. However, it may still be the case that most instances of the problem are easy to manipulate.

A relatively little-known theory of average case complexity exists [28]; that theory introduces the concept of distributional problems, and defines what a reduction between distributional problems is. It is also known that average-case complete problems exist (albeit artificial ones, such as a distributional version of the halting problem).

Sadly, it is very difficult to show that a certain problem is average-case complete, and such results are known only for a handful of problems. Additionally, the goal of the existing theory is to define when a problem is *hard* in the average-case; it does not provide criteria for deciding when a problem is *easy*. A step towards showing that a manipulation is easy on average was made in [8]. It involves an analysis of the plurality protocol with a pre-round, but focuses on a very specific distribution, which does not satisfy some basic desiderata as to what properties an "interesting" distribution should have.

In this paper, we engage in a novel average-case analysis, based on criteria we propose. Coming up with an "interesting" distribution of problem instances with respect to which the average-case complexity is computed is a difficult task, and the solution may be controversial. We analyze problems whose instances are distributed with respect to a *junta distribution*. Such a distribution must satisfy several conditions, which (arguably) guarantee that it focuses on instances that are harder to manipulate. We consider a protocol to be *susceptible* to manipulation when there is a polynomial time algorithm that can usually manipulate it: the probability of failure (when the instances are distributed according to a junta distribution) must be inverse-polynomial. Such an algorithm is known as a *heuristic* polynomial time algorithm.

We use these new methods to show our main result: an important family of protocols, called *scoring* protocols, is susceptible to coalitional manipulation when the number of candidates is constant.[6] Specifically, we contemplate *sensitive* scoring protocols, which include such well-known protocols as Borda and Veto. To accomplish this task, we define a natural distribution μ^* over the instances of a well-defined coalitional manipulation problem, and show that this is a junta distribution. Furthermore, we present the manipulation algorithm GREEDY, and show that it usually succeeds with respect to μ^*.

We also show that all protocols are susceptible to a certain setting of manipulation, where the manipulator is unsure about the others' votes. This result depends upon a basic conjecture regarding junta distributions, but also has implications that transcend our specific definition of these distributions.

In Section 2, we outline some important voting protocols, and properly define the manipulation problems we shall discuss. In Section 3, we formally introduce the tools for our average case analysis: junta distributions, heuristic polynomial time, and susceptibility to manipulations. In Section 4 we present our main result: sensitive scoring protocols are susceptible to coalitional manipulation with few candidates. In Section 5, we discuss the case when a single manipulator is unsure about the other voters' votes. Finally, in Section 6, we present conclusions and directions for future research.

2 Preliminaries

We first describe some common voting protocols and formally define the manipulation problems with which we shall deal. Next, we introduce a useful lemma from probability theory.

2.1 Elections and Manipulations

An election consists of a set C of m candidates, and a set V of n voters, who provide a total order on the candidates. An election also includes a winner determination function from the set of all possible combinations of votes to C. We note that throughout this paper, $m = O(1)$, so the complexity results are in terms of n.

Different voting protocols are distinguished by their winner determination functions. The protocols we shall discuss are:

- *Scoring protocols:* A scoring protocol is defined by vector $\boldsymbol{\alpha}=\langle\alpha_1, \alpha_2, \ldots, \alpha_m\rangle$, such that $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_m$ and $\alpha_i \in \mathbb{N} \cup \{0\}$. A candidate receives α_i points for each voter which ranks it in the i'th place. Examples of scoring protocols are:
  - *Plurality:* $\boldsymbol{\alpha} = \langle 1, 0, \ldots, 0, 0\rangle$.
  - *Veto:* $\boldsymbol{\alpha} = \langle 1, 1, \ldots, 1, 0\rangle$.
  - *Borda:* $\boldsymbol{\alpha} = \langle m-1, m-2, \ldots, 1, 0\rangle$.

[6] Proofs can be found in [21].

- *Copeland:* For each possible pair of candidates, simulate an election; a candidate wins such a pairwise election if more voters prefer it over the opponent. A candidate gets 1 point for each pairwise election it wins, and -1 for each pairwise election it loses.
- *Maximin:* A candidate's score in a pairwise election is the number of voters that prefer it over the opponent. The winner is the candidate whose minimum score over all pairwise elections is highest.
- *Single Transferable Vote (STV):* The election proceeds in rounds. In each round, the candidate's score is the number of voters that rank it highest among the remaining candidates; the candidate with the lowest score is eliminated.

Remark 1. We assume that tie-breaking is always adversarial to the manipulator.[7]

In the case of weighted votes, a voter with weight $k \in \mathbb{N}$ is naturally regarded as k voters who vote unanimously. In this paper, we consider weights in $[0, 1]$. This is equivalent, since any set of integer weights in the range $1, \ldots,$ poly$n$ can be scaled down to weights in the segment $[0, 1]$ with $O(\log n)$ bits of precision.

The main results of the paper focus on scoring protocols. We shall require the following definition:

Definition 1. Let P be a scoring protocol with parameters $\boldsymbol{\alpha} = \langle \alpha_1, \alpha_2, \ldots, \alpha_m \rangle$. We say that P is *sensitive* iff $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_{m-1} > \alpha_m = 0$ (notice the strict inequality on the right).

In particular, Borda and Veto are sensitive scoring protocols.

Remark 2. Generally, from any scoring protocol with $\alpha_{m-1} > \alpha_m$, an equivalent sensitive scoring protocol can be obtained by subtracting α_m on a coordinate-by-coordinate basis from the vector $\boldsymbol{\alpha}$. Moreover, observe that if a protocol is a scoring protocol but is not sensitive, and $\alpha_m = 0$, then $\alpha_{m-1} = 0$. In this case, for three candidates it is equivalent to the plurality protocol, for which most manipulations are tractable even in the worst-case. Therefore, it is sufficient to restrict our results to sensitive scoring protocols.

We next consider some types of manipulations, state the appropriate complexity results, and introduce some notations.

Remark 3. We discuss the constructive *cases, where the goal is trying to make a candidate win, as opposed to* destructive *manipulation, where the goal is to make a candidate lose. Constructive manipulations are always at least as hard (in the worst-case sense) as their destructive counterparts, and in some cases strictly harder (if one is able to determine whether p can be made to win, one can also ask whether any of the other $m-1$ candidates can be made to win, thus making p lose).*

[7] This is a standard assumption, also made, for example, in [6,4]. It does, indeed, make it more straightforward to prove certain results.

Definition 2. In the INDIVIDUAL-MANIPULATION problem, we are given all the other votes, and a preferred candidate p. We are asked whether there is a way for the manipulator to cast its vote so that p wins.

Bartholdi and Orlin [2] show that IM is $\mathcal{NP}$-complete in Single Transferable Vote, provided the number of candidates is unbounded. However, the problem is in $\mathcal{P}$ for most voting schemes, and hence will not be studied here.

Definition 3. In the COALITIONAL-WEIGHTED-MANIPULATION (CWM) problem, we are given a set of weighted votes S, the weights of a set of votes T which have not been cast, and a preferred candidate p. We are asked whether there is a way to cast the votes in T so that p wins the election.

We know [6,4] that CWM is $\mathcal{NP}$-complete in Borda, Veto and Single Transferable Vote, even with 3 candidates, and in Maximin and Copeland with at least 4 candidates.

The CWM version that we shall analyze, which is specifically tailored for scoring protocols, is a slightly modified version whose analysis is more straightforward:

Definition 4. In the SCORING-COALITIONAL-WEIGHTED-MANIPULATION (SCWM) problem, we are given an initial score $S[c]$ for each candidate c, the weights of a set of votes T which have not been cast, and a preferred candidate p. We are asked whether there is a way to cast the votes in T so that p wins the election.

$S[c]$ can be interpreted as c's total score from the votes in S. However, we do not require that there exist a combination of votes that actually induces $S[c]$ for all c.

Definition 5. In the UNCERTAIN-VOTES-WEIGHTED-EVALUATION (UVWE) problem, we are given a weight for each voter, a distribution over all the votes, a candidate p, and a number $r \in [0, 1]$. We are asked whether the probability of p winning is greater than r.

Definition 6. In the UNCERTAIN-VOTES-WEIGHTED-MANIPULATION (UVWM) problem, we are given a single manipulative voter with a weight, weights for all other voters, a distribution over all the others' votes, a candidate p, and a number r, where $r \in [0, 1]$. We are asked whether the manipulator can cast its vote so that p wins with probability greater than r.

If CWM is $\mathcal{NP}$-hard in a protocol, then UVWE and UVWM are also $\mathcal{NP}$-hard in it [6]. These problems will be studied in Section 5. We make the assumption that the given distributions over the others' votes can be sampled in polynomial time.

2.2 Chernoff's Bounds

The following lemma will be of much use later on. Informally, it states that the average of independent identically distributed (i.i.d.) random variables is almost always close to the expectation.

Lemma 1 (Chernoff's Bounds). *Let $X_1, \ldots, X_t$ be i.i.d. random variables such that $a \leq X_i \leq b$ and $\mathrm{E}[X_i] = \mu$. Then for any $\epsilon > 0$, it holds that:*

- $\Pr[\frac{1}{t}\sum_{i=1}^{t} X_i \geq \mu + \epsilon] \leq e^{-2t\frac{\epsilon^2}{(b-a)^2}}$
- $\Pr[\frac{1}{t}\sum_{i=1}^{t} X_i \leq \mu - \epsilon] \leq e^{-2t\frac{\epsilon^2}{(b-a)^2}}$

3 Junta Distributions and Susceptible Mechanisms

In this section we lay the mathematical foundations required for an average-case analysis of the complexity of manipulations. All of the definitions are as general as possible; they can be applied to the manipulation of any mechanism, not merely to the manipulation of voting protocols.

We describe a distribution over the instances of a problem as a collection of distributions $\mu_1, \ldots, \mu_n, \ldots$, where μ_n is a distribution over the instances x such that $|x| = n$. We wish to analyze problems whose instances are distributed with respect to a distribution which focuses on hard-to-manipulate instances. Ideally, we would like to insure that if one manages to produce an algorithm which can usually manipulate instances according to this distinguished "difficult" distribution, the algorithm would also usually succeed when the instances are distributed with respect to most other reasonable distributions.

Definition 7. Let $\mu = \{\mu_n\}_{n\in\mathbb{N}}$ be a distribution over the possible instances of an $\mathcal{NP}$-hard manipulation problem M. μ is a *junta* distribution if and only if μ has the following properties:

1. Hardness: The restriction of M to μ is the manipulation problem whose possible instances are only:

$$\bigcup_{n\in\mathbb{N}} \{x : |x| = n \wedge \mu_n(x) > 0\}.$$

   Deciding this restricted problem is still $\mathcal{NP}$-hard.
2. Balance: There exist a constant $c > 1$ and $N \in \mathbb{N}$ such that for all $n \geq N$:

$$\frac{1}{c} \leq \Pr_{x\sim\mu_n}[M(x) = 1] \leq 1 - \frac{1}{c}.$$

3. Dichotomy: for all n and instances x such that $|x| = n$:

$$\mu_n(x) \geq 2^{-\mathrm{poly}n} \vee \mu_n(x) = 0.$$

If M is a voting manipulation problem, we also require the following property:

4. Symmetry: Let v be a voter whose vote is given, let $c_1, c_2 \neq p$ be two candidates, and let $i \in \{1, \ldots, m\}$. The probability that v ranks c_1 in the i'th place is the same as the probability that v ranks c_2 in the i'th place.

If M is a coalitional manipulation problem, we also require the following property:

5. Refinement: Let x be an instance such that $|x| = n$ and $\mu_n(x) > 0$; if all colluders voted identically, then p would not be elected.

The name "junta distribution" comes from the idea that in such a distribution, relatively few "powerful" and difficult instances represent all the other problem instances. Alternatively, our intent is to have a few problematic distributions (the family of junta distributions) convincingly represent all other distributions with respect to the average-case analysis.

The first three properties are basic, and are relevant to problems of manipulating any mechanism. The definition is modular, and additional properties may be added on top of the basic three, in case one wishes to analyze a mechanism which is not a voting protocol.

The exact choice of properties is of extreme importance (and, as we mentioned above, may be arguable). We shall briefly explain our choices. Hardness is meant to insure that the junta distribution contains hard instances. Balance guarantees that a trivial algorithm which always accepts (or always rejects) has a significant chance of failure. The dichotomy property helps in preventing situations where the distribution gives a (positive but) negligible probability to all the hard instances, and a high probability to several easy instances.

We now examine the properties that are specific to manipulation problems. The necessity of symmetry is best explained by an example. Consider CWM in STV with $m \geq 3$. One could design a distribution where p wins if and only if a distinguished candidate loses the first round. Such a distribution could be tailored to satisfy the other conditions, but misses many of the hard instances. In the context of SCWM, we interpret symmetry in the following way: for every two candidates $c_1, c_2 \neq p$ and $y \in \mathbb{R}$,

$$\Pr_{x \sim \mu_n}[S[c_1] = y] = \Pr_{x \sim \mu_n}[S[c_2] = y].$$

Refinement is less important than the other four properties, but seems to help in concentrating the probability on hard instances. Observe that refinement is only relevant to coalitional manipulation; we believe that in the analysis of individual voting manipulation problems, the first four properties are sufficient.

Definition 8. [28] A *distributional problem* is a pair $\langle L, \mu \rangle$ where L is a decision problem and μ is a distribution over the set $\{0,1\}^*$ of possible inputs.

Informally, an algorithm is a heuristic polynomial time algorithm for a distributional problem if it runs in polynomial time, and fails only on a small fraction of the inputs. We now give a formal definition; this definition is inspired by [28] (there the same name is used for a somewhat different definition).

Definition 9. Let M be a manipulation problem and let $\langle M, \mu \rangle$ be a distributional problem.

1. An algorithm A is a *deterministic heuristic polynomial time* algorithm for the distributional manipulation problem $\langle M, \mu \rangle$ if A always runs in polynomial time, and there exists a polynomial p and $N \in \mathbb{N}$ such that for all $n \geq N$:

$$\Pr_{x \sim \mu^n}[A(x) \neq M(x)] < \frac{1}{p(n)}. \tag{1}$$

2. Let A be a probabilistic algorithm, which uses a random string s. A is a *probabilistic heuristic polynomial time* algorithm for the distributional manipulation problem $\langle M, \mu \rangle$ if A always runs in polynomial time, and there exists a polynomial p and $N \in \mathbb{N}$ such that for all $n \geq N$:

$$\Pr_{x \sim \mu^n, s}[A(x) \neq M(x)] < \frac{1}{p(n)}. \tag{2}$$

Probabilistic algorithms have two potential sources of failure: an unfortunate choice of input, or an unfortunate choice of random string s. The success or failure of deterministic algorithms depends only on the choice of input.

We now combine all the definitions introduced in this section in an attempt to establish when a mechanism is susceptible to manipulation in the average case. The following definition abuses notation a bit: M is both used to refer to the manipulation itself, and the corresponding decision problem.

Definition 10. We say that a mechanism is *susceptible* to a manipulation M if there exists a junta distribution μ, such that there exists a deterministic/probabilistic heuristic polynomial time algorithm for $\langle M, \mu \rangle$.

4 Susceptibility to SCWM

Recall [6,4] that in Borda and Veto, CWM is $\mathcal{NP}$-hard, even with 3 candidates. Since Borda and Veto are examples of sensitive scoring protocols, we would like to know how resistant this family of protocols really is with respect to coalitional manipulation. In this section we use the methods from the previous section to present our main result:

Theorem 1. *Let P be a sensitive scoring protocol. Then P, with candidates $C = \{p, c_1, \ldots, c_m\}$, $m = O(1)$, is susceptible to SCWM.*

Intuitively, the instances of CWM (or SCWM) which are hard are those that require a very specific partitioning of the voters in T to subsets, where each subset votes unanimously. These instances are rare in any reasonable distribution; this insight will ultimately yield the theorem.

The following proposition generalizes Theorem 1 of [6] and Theorem 2 of [4], and justifies our focus on the family of sensitive scoring protocols. A stronger version of Proposition 1 has been independently proven in [16].

Proposition 1. *Let P be a sensitive scoring protocol. Then CWM in P is $\mathcal{NP}$-hard, even with 3 candidates.*

Definition 11. In the PARTITION problem, we are given a set of integers $\{k_i\}_{i\in[t]}$, summing to $2K$, and are asked whether a subset of these integers sum to K.

It is well-known that PARTITION is $\mathcal{NP}$-complete.

Proof (of Proposition 1). All proofs in the paper are omitted, but can be seen in [21].

Since an instance of CWM can be translated to an instance of SCWM in the obvious way, we have:

Corollary 1. *Let P be a sensitive scoring protocol. It holds that SCWM in P is $\mathcal{NP}$-hard, even with 3 candidates.*

4.1 A Junta Distribution

Let $w(v)$ denote the weight of voter v, and let W denote the total weight of the votes in T; P is a sensitive scoring protocol. We denote $|T| = n$: the size of T is the size of the instance.

Consider a distribution $\mu^* = \{\mu_n^*\}_{n\in\mathbb{N}}$ over the instances of CWM in P, with $m+1$ candidates $p, c_1, \ldots, c_m$, where each μ_n^* is induced by the following sampling algorithm:

1. $\forall v \in T$: Randomly and independently choose $w(v) \in [0,1]$ (up to $O(\log n)$ bits of precision).
2. $\forall i \in \{1,\ldots,m\}$: Randomly and independently choose $S[c_i] \in [(\alpha_1-\alpha_2)W, \alpha_1 W]$ (up to $O(\log n)$ bits of precision).

We assume that $S[p] = 0$, i.e., all voters in S rank p last. This assumption is not a restriction. If it holds for a candidate c that $S[c] \leq S[p]$, then candidate c will surely lose, since the colluders all rank p first. Therefore, if $S[p] > 0$, we may simply normalize the scores by subtracting $S[p]$ from the scores of all candidates. This is equivalent to our assumption.

Remark 4. We believe that μ^* is the most natural distribution with respect to which coalitional manipulation in scoring protocols should be studied. Even if one disagrees with the exact definition of junta distribution, μ^* should satisfy many reasonable conditions one could produce.

We shall, of course, (presently) show that the distribution possesses the properties of a junta distribution.

Proposition 2. *Let P be a sensitive scoring protocol. Then μ^* is a junta distribution for SCWM in P with $C = \{p, c_1, \ldots, c_m\}$, and $m = O(1)$.*

4.2 A Heuristic Polynomial Time Algorithm

We now present our algorithm GREEDY for SCWM, given as Algorithm 1. $\boldsymbol{w}$ denotes the vector of the weights of voters in $T = \{t_1, \ldots, t_n\}$.

Algorithm 1. Decides SCWM

procedure GREEDY($S, \boldsymbol{w}, p$)
  for all $c \in C$ **do** ▷ Initialization
    $S_0[c] \leftarrow S[c]$
  end for
  for $i = 1$ to n **do** ▷ All voters in T
    Let $j_1, j_2, \ldots, j_m$ s.t. $\forall l,\ S_{i-1}[c_{j_{l-1}}] \leq S_{i-1}[c_{j_l}]$
    Voter t_i votes $p \succ c_{j_1} \succ c_{j_2} \succ \ldots \succ c_{j_m}$
    for $l = 1$ to m **do** ▷ Update score
      $S_i[c_{j_l}] \leftarrow S_{i-1}[c_{j_l}] + w(t_i)\alpha_{l+1}$
    end for
    $S_i[p] \leftarrow S_{i-1}[p] + w(t_i)\alpha_1$
  end for
  if $\operatorname{argmax}_{c \in C} S_n[c] = \{p\}$ **then** ▷ p wins
    return true
  else
    return false
  end if
end procedure

The voters in T, according to some order, each rank p first, and the rest of the candidates by their current score: the candidate with the lowest current score is ranked highest. GREEDY accepts if and only if p wins this election.

This algorithm, designed specifically for scoring protocols, is a realization of an abstract greedy algorithm: at each stage, voter t_i ranks the undesirable candidates in an order that minimizes the highest score that any undesirable candidate obtains after the current vote. If there is a tie between several permutations, the voter chooses the option such that the second highest score is as low as possible, etc. In any case, every colluder always ranks p first.

Remark 5. This abstract scheme might also be appropriate for protocols such as Maximin and Copeland. Similarly to scoring protocols, in these two protocols the colluders are always better off by ranking p first. In addition, the abstract greedy algorithm can be applied to Maximin and Copeland since the result of an election is based on the score each candidate has (unlike STV, for example).

In the following lemmas, a *stage* in the execution of the algorithm is an iteration of the for loop.

Lemma 2. *If there exists a stage i_0 during the execution of* GREEDY*, and two candidates $a, b \neq p$, such that*

$$|S_{i_0}[a] - S_{i_0}[b]| \leq \alpha_2, \tag{3}$$

then for all $i \geq i_0$ it holds that $|S_i[a] - S_i[b]| \leq \alpha_2$.

Lemma 3. *Let $p \neq a, b \in C$, and suppose that there exists a stage i_0 such that $S_{i_0}[a] \geq S_{i_0}[b]$, and a stage $i_1 \geq i_0$ such that $S_{i_1}[b] \geq S_{i_1}[a]$. Then for all $i \geq i_1$ it holds that $|S_i[a] - S_i[b]| \leq \alpha_2$.*

Lemma 4. *Let P be a sensitive scoring protocol, and assume* Greedy *errs on an instance of SCWM in P which has a successful manipulation. Then there is $d \in \{2, 3, \ldots, m\}$, and a subset of candidates $D = \{c_{j_1}, \ldots, c_{j_d}\}$, such that:*

$$\begin{aligned} &\sum_{i=1}^{d}(\alpha_1 W - S[c_{j_i}]) - \sum_{i=1}^{d-1}(i \cdot \alpha_2) \leq W \sum_{i=1}^{d} \alpha_{m+2-i} \\ &\leq \sum_{i=1}^{d}(\alpha_1 W - S[c_{j_i}]). \end{aligned} \tag{4}$$

Lemma 5. *Let M be SCWM in a sensitive scoring protocol P with $C = \{p, c_1, \ldots, c_m\}$, m=O(1). Then* Greedy *is a deterministic heuristic polynomial time algorithm for $\langle M, \mu^* \rangle$.*

Clearly, Theorem 1 directly follows.

5 Susceptibility to UVWM

In this section we shall show:

Theorem 2. *Let P be a voting protocol such that there exists a junta distribution μ^P over the instances of UVWM in P, with the following property: r is uniformly distributed in $[0, 1]$. Then P, with candidates $C = \{p, c_1, \ldots, c_m\}$, $m = O(1)$, is susceptible to UVWM.*

The existence of a junta distribution with r uniformly distributed is a very weak requirement (it is even quite natural to have r uniformly distributed). In fact, the following claim is very likely to be true:

Conjecture 1. Let P be a voting protocol. Then there exists a junta distribution μ^P over the instances of UVWM in P, with r uniformly distributed in $[0, 1]$.

If this conjecture is indeed true, we have that all voting protocols are susceptible to UVWM. If for some reason the conjecture is not true with respect to our definition of junta distributions, then perhaps the definition is too restrictive and should be modified accordingly.

To prove Theorem 2, we require a procedure named Sample, which decides UVWE. Sample samples the given distribution on the votes n^3 times, and calculates the winner of the election each time. If p won more than an r-fraction of the elections then the procedure accepts, otherwise it rejects. We omit the details of the procedure.

Algorithm 2. Decides UVWM

```
1: procedure SAMPLE-AND-MANIPULATE(w, ν, p, r)
2:     for all permutations of the m + 1 candidates do
3:         π ← next permutation
4:         ν* ← the manipulator votes π
5:                                   ▷ others' votes are always distributed w.r.t. ν
6:         if SAMPLE(w, ν*, p, r) then
7:             return true
8:         end if
9:     end for
10:    return false
11: end procedure
```

Lemma 6. *Let P be a voting protocol, and E be UVWE in P with $C = \{p, c_1, \ldots, c_m\}$. Furthermore, let μ be a distribution over the instances of E, with r uniformly distributed in $[0, 1]$. Then there exists N such that for all $n \geq N$:*

$$\Pr_{x \sim \mu_n} [\text{SAMPLE}(x) \neq E(x)] \leq \frac{1}{\text{poly}n}.$$

We now present an algorithm, SAMPLE-AND-MANIPULATE that decides UVWM; it is given as Algorithm 2. Here, $\boldsymbol{w}$ denotes the weights of all voters including the manipulator, and ν is the given distribution over the others' votes.

Given an instance of UVWM, SAMPLE-AND-MANIPULATE generates $(m+1)!$ instances of the UVWE problem, one for each of the manipulator's possible votes, and executes SAMPLE on each instance. SAMPLE-AND-MANIPULATE accepts if and only if SAMPLE accepts one of the instances.

Lemma 7. *Let P be a voting protocol, and M be UVWM in P with $C = \{p, c_1, \ldots, c_m\}$, $m = O(1)$. Furthermore, let μ be a distribution over the instances of UVWM, with r uniformly distributed in $[0, 1]$. It holds that* SAMPLE-AND-MANIPULATE *is a probabilistic heuristic polynomial time algorithm for $\langle M, \mu \rangle$.*

6 Future Research

The issue of resistance of mechanisms to manipulation is important, particularly in the context of voting protocols. Most results on this issue use $\mathcal{NP}$-hardness as the complexity measure. One of this paper's main contributions has been introducing tools that can be utilized in showing that manipulating mechanisms is *easy* in the average case. We were concerned with the likely case of coalitional manipulation, and showed that sensitive scoring protocols are susceptible to such manipulation when the number of candidates is constant.

These results suggest that scoring protocols cannot be safely employed. More importantly, this paper should be seen as a starting point for studying the average case complexity of other types of manipulations, in other protocols. In addition, the definitions in Section 3 are deliberately general, and can be applied to

manipulations of mechanisms that are not voting mechanisms. One such mechanism of which we are aware, whose manipulation is $\mathcal{NP}$-hard, is presented in [1].

There is still room for debate as to the exact definition of a junta distribution, especially if Conjecture 1 turns out to be false. It may also be the case that there are "unconvincing" distributions that satisfy all of the (current) conditions of a junta distribution. It might prove especially fruitful to show that a heuristic polynomial time algorithm with respect to a junta distribution also has the same property with respect to some easy distributions, such as the uniform distribution.

An issue of great importance is coming up with natural criteria to decide when a manipulation problem is *hard* in the average-case. The traditional definition of average-case completeness is very difficult to work with in general; is there a satisfying definition that applies specifically to the case of manipulations? Once the subject is more fully understood, this understanding can be used to design mechanisms that are hard to manipulate in the average-case.

Acknowledgment

This work was partially supported by grant #039-7582 from the Israel Science Foundation.

References

1. Yoram Bachrach and Jeffrey S. Rosenschein. Achieving allocatively-efficient and strongly budget-balanced mechanisms in the network flow domain for bounded-rational agents. In *The Nineteenth International Joint Conference on Artificial Intelligence*, pages 1653–1654, Edinburgh, Scotland, August 2005. Full version published in The Seventh International Workshop on Agent-Mediated Electronic Commerce: Designing Mechanisms and Systems (AMEC 2005), Utrecht, The Netherlands, July 2005.
2. J. Bartholdi and J. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8(4):341–354, 1991.
3. J. Bartholdi, C. A. Tovey, and M. A. Trick. How hard is it to control an election. *Mathematical and Computer Modelling*, 16:27–40, 1992.
4. V. Conitzer, J. Lang, and T. Sandholm. How many candidates are needed to make elections hard to manipulate? In *Proceedings of the International Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 201–214, Bloomington, Indiana, 2003.
5. V. Conitzer and T. Sandholm. Complexity of manipulating elections with few candidates. In *Proceedings of the National Conference on Artificial Intelligence*, pages 314–319, Edmonton, Canada, July 2002.
6. V. Conitzer and T. Sandholm. Complexity of manipulating elections with few candidates. In *Proceedings of the National Conference on Artificial Intelligence*, pages 314–319, Edmonton, Canada, July 2002.
7. V. Conitzer and T. Sandholm. Universal voting protocol tweaks to make manipulation hard. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 781–788, Acapulco, Mexico, August 2003.

8. E. Elkind and H. Lipmaa. Small coalitions cannot manipulate voting. In *International Conference on Financial Cryptography*, Lecture Notes in Computer Science. Springer-Verlag, Roseau, The Commonwealth of Dominica, 2005.
9. Eithan Ephrati and Jeffrey S. Rosenschein. The Clarke Tax as a consensus mechanism among automated agents. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 173–178, Anaheim, California, July 1991.
10. Eithan Ephrati and Jeffrey S. Rosenschein. A heuristic technique for multiagent planning. *Annals of Mathematics and Artificial Intelligence*, 20:13–67, Spring 1997.
11. Robert M. Gagné and Karen L. Medsker. *The Conditions of Learning Training Applications.* Harcourt Brace & Company, 1996.
12. S. Ghosh, M. Mundhe, K. Hernandez, and S. Sen. Voting for movies: the anatomy of a recommender system. In *Proceedings of the Third Annual Conference on Autonomous Agents*, pages 434–435, 1999.
13. A. Gibbard. Manipulation of voting schemes. *Econometrica*, 41:587–602, 1973.
14. T. Haynes, S. Sen, N. Arora, and R. Nadella. An automated meeting scheduling system that utilizes user preferences. In *Proceedings of the First International Conference on Autonomous Agents*, pages 308–315, 1997.
15. E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Anyone but him: The complexity of precluding an alternative. In *Proceedings of the 20th National Conference on Artificial Intelligence*, Pittsburgh, July 2005.
16. E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. University of Rochester Department of Computer Science Technical Report 861, 2005.
17. William G. Hoover. *Computational Statistical Mechanics.* Elsevier, 1991.
18. George Johnson. All science is computer science, 25 March 2001. The New York Times, Week in Review, pages 1, 5.
19. Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.
20. Ariel D. Procaccia and Jeffrey S. Rosenschein. The distortion of cardinal preferences in voting. In *The Tenth International Workshop on Cooperative Information Agents (CIA 2006)*, Edinburgh, September 2006.
21. Ariel D. Procaccia and Jeffrey S. Rosenschein. Junta distributions and the average-case complexity of manipulating elections. In *The Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 497–504, Hakodate, Japan, May 2006.
22. Ariel D. Procaccia, Jeffrey S. Rosenschein, and Aviv Zohar. Multi-winner elections: Complexity of manipulation, control and winner-determination. In *The Eighth International Workshop on Agent-Mediated Electronic Commerce (AMEC 2006)*, pages 15–28, Hakodate, Japan, May 2006.
23. Jeffrey S. Rosenschein and Michael R. Genesereth. Deals among rational agents. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 91–99, Los Angeles, California, August 1985.
24. T. Sandholm and V. Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS-95)*, pages 328–335, San Francisco, 1995.
25. M. Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10:187–217, 1975.
26. Moshe Tennenholtz and Alon Altman. On the axiomatic foundations of ranking systems. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI'05)*, pages 917–922, Edinburgh, August 2005.

27. Jeffrey D. Thomas, Taesik Lee, and Nam P. Suh. A function-based framework for understanding biological systems. *Annual Review of Biophysics and Biomolecular Structure*, 33:75–93, 2004.
28. L. Trevisan. Lecture notes on computational complexity. Available from http://www.cs.berkeley.edu/~luca/notes/complexitynotes02.pdf, 2002. Lecture 12.
29. Michael P. Wellman. The economic approach to artificial intelligence. *ACM Computing Surveys*, 27:360–362, 1995.