Nonexistence of Voting Rules That Are Usually Hard to Manipulate

Vincent Conitzer and **Tuomas Sandholm**
Carnegie Mellon University
Computer Science Department
5000 Forbes Avenue, Pittsburgh, PA 15213
{conitzer, sandholm}@cs.cmu.edu

Abstract

Aggregating the preferences of self-interested agents is a key problem for multiagent systems, and one general method for doing so is to vote over the alternatives (candidates). Unfortunately, the Gibbard-Satterthwaite theorem shows that when there are three or more candidates, all reasonable voting rules are manipulable (in the sense that there exist situations in which a voter would benefit from reporting its preferences insincerely). To circumvent this impossibility result, recent research has investigated whether it is possible to make finding a beneficial manipulation computationally hard. This approach has had some limited success, exhibiting rules under which the problem of finding a beneficial manipulation is NP-hard, #P-hard, or even PSPACE-hard. Thus, under these rules, it is unlikely that a computationally efficient algorithm can be constructed that *always* finds a beneficial manipulation (when it exists). However, this still does not preclude the existence of an efficient algorithm that *often* finds a successful manipulation (when it exists). There have been attempts to design a rule under which finding a beneficial manipulation is *usually* hard, but they have failed. To explain this failure, in this paper, we show that it is in fact impossible to design such a rule, if the rule is also required to satisfy another property: a large fraction of the manipulable instances are both weakly monotone, and allow the manipulators to make either of exactly two candidates win. We argue why one should expect voting rules to have this property, and show experimentally that common voting rules clearly satisfy it. We also discuss approaches for potentially circumventing this impossibility result.

Introduction

In multiagent settings, it is of central importance to be able to aggregate the preferences of multiple agents to arrive at a joint decision. One general method for doing so is to let the agents *vote* over the alternatives (*candidates*). Here, the candidates can be potential political representatives (as in a presidential election), but they can also be joint plans, allocations of tasks or resources, *etc*. One problem that can occur is strategic voting (*manipulation*) where one or more voters do not rank the candidates according to their true preferences, but rather in such a way as to make the outcome more favorable to themselves. This may lead to the choice of an outcome that is good with respect to the voters' reported preferences, but bad with respect to their true preferences. Unfortunately, a result known as the Gibbard-Satterthwaite theorem (Gibbard 1973; Satterthwaite 1975) shows that when there are at least three candidates, every nondictatorial rule for choosing the winning candidate is vulnerable to manipulation, in the sense that there are some preferences of the voters so that some voter is better off voting strategically.

In recent years, an approach that has been pursued to circumvent this impossibility result is to try to make manipulation *computationally* hard. The idea is that it does not matter if a beneficial manipulation exists if the manipulators do not have the computational power to find it. This idea was first investigated in the social choice literature, where it was shown that two rules are NP-hard to manipulate even by an individual manipulator, if the number of candidates is unbounded (Bartholdi, Tovey, & Trick 1989a; Bartholdi & Orlin 1991). In recent years, this idea has been picked up in the AI literature. It has been shown that manipulation by a coalition of weighted voters can be NP-hard even when there are few candidates (Conitzer & Sandholm 2002; Conitzer, Lang, & Sandholm 2003). It has also been shown that adding a preround can make voting rules up to PSPACE-hard to manipulate even by an individual, even when the original rule is easy to manipulate (Conitzer & Sandholm 2003). More generally, it has been shown that *hybridizing* voting rules with each other can make manipulation harder (Elkind & Lipmaa 2005a).

One weakness that all of these results have in common is that they only show *worst-case* hardness. That is, the results show that it is unlikely that an efficient algorithm can be designed that finds a beneficial manipulation in *all* instances for which a beneficial manipulation exists. However, this does not mean that there do not exist efficient manipulation algorithms that find a beneficial manipulation in *many* instances. If such algorithms do in fact exist, then computational hardness constitutes a leaky barrier to manipulation at best (though it is presumably still better than nothing).

A truly satisfactory solution to the problem would be to have a rule that is hard to manipulate in *all* instances. However, this is too much to ask for: for example, a manipulation algorithm could have a small database of instances with precomputed solutions, and it would merely need to check against this database to successfully manipulate some instances. Still, we may ask whether it is possible to make (say) 99% of instances hard to manipulate. It is generally agreed that this would have a much greater impact on the design of voting rules in practice than merely worst-case hardness (Conitzer, Lang, & Sandholm 2003; Elkind & Lipmaa 2005b), but none of the multiple efforts to achieve this objective have succeeded.

In this paper, we present an impossibility result that makes it seem unlikely that such an objective can be achieved by any reasonable voting rule. This is not the first such impossibility result: a previous result (Procaccia & Rosenschein 2006) shows that a specific subclass of voting rules is usually easy to manipulate when the number of candidates is constant and a specific distribution over instances is used (where the distribution is chosen to have certain properties that would appear to make manipulation more difficult). By contrast, our result does not require any restriction on the voting rule, number of candidates, or distribution over instances. Our result states that a voting rule/instance distribution pair cannot simultaneously be usually hard to manipulate, and have certain natural properties (which depend on the rule and distribution).

Definitions

Voting rules

In a voting setting, we have a set of m candidates, C. Every voter i ($1 \leq i \leq n$) casts a *vote*, which consists of a ranking r_i of all the candidates. We use the notation $c_1 \succ_{r_i} c_2$ to denote that c_1 is ranked ahead of c_2 in vote r_i. A *voting rule* is a mapping that takes as input a vector of votes (one for each voter), and produces as output a single candidate (the winner). We will consider the following common rules:

• *Scoring rules.* Let $\vec{\alpha} = \langle \alpha_1, \ldots, \alpha_m \rangle$ be a vector of integers. For each vote, a candidate receives α_1 points if it is ranked first in the vote, α_2 if it is ranked second, *etc.* The candidate with the most points wins. The *Borda* rule is the scoring rule with $\vec{\alpha} = \langle m-1, m-2, \ldots, 0 \rangle$. The *plurality* rule is the scoring rule with $\vec{\alpha} = \langle 1, 0, \ldots, 0 \rangle$. The *veto* rule is the scoring rule with $\vec{\alpha} = \langle 1, 1, \ldots, 1, 0 \rangle$.

• *Single transferable vote (STV).* This rule proceeds through a series of $m - 1$ rounds. In each round, the candidate with the lowest plurality score (that is, the fewest votes ranking it first among the remaining candidates) is eliminated (and each of the votes for that candidate "transfers" to the next remaining candidate in the order given in that vote). The winner is the last remaining candidate.

• *Plurality with run-off.* In this rule, a first round eliminates all candidates except the two with the highest plurality scores. Votes are transferred to these as in the STV rule, and a second round determines the winner from these two.

• *Maximin* (aka. *Simpson*). For any two candidates i and j, let $N(i, j)$ be the number of votes that prefer i to j. The *maximin score* of i is $s(i) = \min_{j \neq i} N(i, j)$—that is, i's worst performance in a *pairwise election*.

• *Copeland.* For any two candidates i and j, let $C(i, j) = 1$ if $N(i, j) > N(j, i)$, $C(i, j) = 1/2$ if $N(i, j) = N(j, i)$, and $C(i, j) = 0$ if $N(i, j) < N(j, i)$. The *Copeland score* of candidate i is $s(i) = \sum_{j \neq i} C(i, j)$.

• *Bucklin.* For any candidate i and integer l, let $B(i, l)$ be the number of votes that rank candidate i among the top l candidates. The winner is $\arg\min_i(\min\{l : B(i, l) > n/2\})$. That is, if we say that a voter "approves" its top l candidates, then we repeatedly increase l by 1 until some candidate is approved by more than half the voters, and this candidate wins.

Sometimes votes are *weighted*; a vote of weight K counts as K votes of weight 1. All of the rules above allow for ties between candidates, so it is necessary to specify some tiebreaking rule. In this paper, we will simply assume that ties are always broken in favor of the lower-indexed candidate (although our results do not depend on this).

We will also consider the *Condorcet* criterion, which states that if there is a candidate i that is preferred to any other candidate by a majority of the voters (that is, $N(i, j) > N(j, i)$ for all $j \neq i$), then candidate i wins. Note that this is a criterion, not a rule: in many instances, there is no such candidate i, in which case the Condorcet criterion does not specify which candidate should win. The Condorcet criterion is perhaps the best-known criterion that voting rules are sometimes expected to satisfy; nevertheless, only a few of the above rules satisfy it (maximin and Copeland).

Manipulation

The computational problem of manipulation has been defined in various ways, but typical definitions are special cases of the following general problem: given the nonmanipulators' votes, can the manipulator(s) cast their votes in such a way that one candidate from a given set of preferred candidates wins? In this paper, we study a more difficult manipulation problem: we require that the manipulator(s) find the set of *all* the candidates that they can make win (as well as votes that will bring this about). This stronger requirement makes our impossibility result stronger: we will show that even this more powerful type of manipulation cannot be prevented.

Definition 1 *A* manipulation instance *is given by a voting rule R, a vector of* nonmanipulator votes *$v = (r_1^{NM}, \ldots, r_n^{NM})$, a vector of* weights *$v^s = (d_1^{NM}, \ldots, d_n^{NM})$ for the nonmanipulators, and a vector of weights $w^s = (d_1^M, \ldots, d_k^M)$ for the manipulators. A manipulation algorithm* succeeds *on this instance if it produces a set of pairs $\{(w_{i_1}, c_{i_1}), \ldots, (w_{i_q}, c_{i_q})\}$ such that 1) if the manipulators cast the vector of votes w_{i_j}, then c_{i_j} wins, and 2) if a candidate c does not occur in this set as one of the c_{i_j}, then there is no vector of manipulator votes w that makes c win.*

An instance is *manipulable* if the manipulators can make more than one candidate win. Nonmanipulable instances are easy to solve: any algorithm that is sound (in that it does not produce incorrect (w_{i_j}, c_{i_j}) pairs) and that returns at least one (w_{i_j}, c_{i_j}) pair (which is easy to do, by simply checking what the rule will produce for a given vector of votes) will succeed. Hence, we focus on manipulable instances only.

To investigate whether voting rules are *usually* easy to manipulate, we also need a probability distribution over (manipulable) instances. Our impossibility result does not require a specific distribution, but in the experimental section of the paper, we study a specific family of distributions.

Weak monotonicity

Informally, a rule is *monotone* if ranking a candidate higher never hurts that candidate. All the rules mentioned before, with the exceptions of STV and plurality with runoff, are monotone. We now formally define a weak notion of monotonicity that is implied by (but does not imply) standard notions of monotonicity. (We omit the details of the implications due to space constraint.) We note that we want our definition of monotonicity to be as weak as possible so that our impossibility result will be as strong as possible. We define when an *instance* is weakly monotone, so that even rules that are not (everywhere) weakly monotone can be (and typically are) weakly monotone on most instances.

Definition 2 *We say that a voting rule R is* weakly monotone *for manipulators with weights w^s and nonmanipulator votes v with weights v^s if for every pair of candidates c_1, c_2, one of the following conditions holds: 1) c_2 does not win for any manipulator votes; or 2) if all the manipulators rank c_2 first and c_1 last, then c_1 does not win.*

Impossibility result

We now present an algorithm that seeks to identify two candidates that can be made to win. The algorithm is universal in that it does not depend on the voting rule used, except for the places in which it calls the rule as a (black-box) subroutine.

```
Find-Two-Winners($R, C, v, v^s, w^s$)
choose an arbitrary manipulator vote vector $w_1$
$c_1 \leftarrow R(C, v, v^s, w_1, w^s)$
for every $c_2 \in C, c_2 \neq c_1$ {
  choose $w_2$ in which every vote ranks $c_2$ first and $c_1$ last
  $c \leftarrow R(C, v, v^s, w_2, w^s)$
  if $c_1 \neq c$ return $\{(w_1, c_1), (w_2, c)\}$ }
return $\{(w_1, c_1)\}$
```

If voting rule R can be executed in polynomial time, then so can Find-Two-Winners.

Theorem 1 Find-Two-Winners *will succeed on every instance that both a) is weakly monotone and b) allows the manipulators to make either of exactly two candidates win.*

Proof: Find-Two-Winners is sound in the sense that it will never output a manipulation that is incorrect. It will certainly find one candidate that the manipulators can make win (c_1). Thus, we merely need to show that it will find the second candidate that can be made to win; let us refer to this candidate as c'. If the algorithm reaches the iteration of the **for** loop in which $c_2 = c'$, then in this round, either $c \neq c_1$, in which case we must have $c = c'$ because there are no other candidates that can be made to win, or $c = c_1$. But in the latter case, we must conclude that $c_2 = c'$ cannot be made to win, due to weak monotonicity—which is contrary to assumption. Hence it must be that $c = c'$. If the algorithm does not reach the iteration of the **for** loop in which $c_2 = c'$, it must have found a manipulation that produced a winner other than c_1 in an earlier iteration, and (by assumption) this other winner can only be c'. ■

It is not possible to extend the algorithm so that it also succeeds on all weakly monotone instances in which *three* candidates can be made to win. When three candidates can be made to win, even under monotone rules, it is possible that one of these candidates can only win if some manipulators vote differently from the other manipulators. In fact, the problem of deciding whether multiple weighted manipulators can make a given candidate win is NP-complete, even when there are only three candidates and the Borda or veto rule is used (both of which are monotone rules) (Conitzer & Sandholm 2002; Conitzer, Lang, & Sandholm 2003). Any algorithm that does succeed on all weakly monotone instances in which at most three candidates can be made to win would be able to solve this NP-complete problem, and thus cannot run in polynomial time (or it would show that P = NP).

The impossibility result now follows as a corollary.

Corollary 1 *For any $p \in [0, 1]$, there does not exist any combination of an efficiently executable voting rule R and a distribution d over instances such that*

1. *the probability of drawing an instance that is both a) weakly monotone, and b) such that either of exactly two candidates can be made to win, is at least p; and*
2. *for any computationally efficient manipulation algorithm, the probability that an instance is drawn on which the algorithm succeeds is smaller than p.*

This impossibility result is relevant only insofar as one expects a voting rule to satisfy Property 1 in the corollary (high probability of drawing a weakly monotone instance in which either of exactly two candidates can be made to win). Before we argue why one should in fact expect this, it is helpful to consider how a skeptic might argue that an impossibility result such as this one is irrelevant. At a minimum, the skeptic should argue that one of the properties required by the result is not sufficiently desirable or necessary to insist on it. The skeptic could make her case much stronger by actually exhibiting a voting rule that satisfies all properties except for the disputed one, and that still seems intuitively desirable. (For example, Arrow's impossibility result (Arrow 1963) is often criticized on the basis that the *independence of irrelevant alternatives* property is unnecessarily strong, and this is the only property that common voting rules fail to satisfy.)

Conversely, we will first argue directly for the desirability of Property 1 in Corollary 1. We will then provide indirect evidence that it will be difficult to construct a sensible rule that does not satisfy this property, by showing experimentally that all common rules satisfy it very strongly (that is, a large fraction of manipulable instances are weakly monotone and such that only two candidates can be made to win).

Arguing directly for Property 1

In this section, we argue why one should expect many manipulable instances to be both a) weakly monotone and b) such that the manipulator(s) can make either of exactly two candidates win. We first make a simple observation: if manipulable instances are usually weakly monotone, and they usually allow the manipulator(s) to make either of exactly two candidates win, then a significant fraction of manipulable instances have both of properties a) and b). More precisely:

Proposition 1 *If the probability of drawing a weakly monotone instance is p, and the probability of drawing an instance in which either of exactly two candidates can be made to win is q, then the probability of drawing an instance with both properties is at least $p + q - 1$.*

Proof: The remaining proofs are omitted to save space. ■

With this in mind, we will now argue separately for each of the two properties a) and b).

The argument for Property a)—most manipulable instances should be weakly monotone—is easy to make. The reason is that if the manipulators rank certain candidates higher, this should, in general, benefit those candidates. If this were not the case, then the manipulators' votes would lose their natural interpretation that they support certain candidates over others, and we are effectively asking the manipulators to submit a string of bits without any inherent meaning. (Incidentally, if this does not bother us, it is easy to design rules that are always hard to manipulate: for example, we can count an agent's vote only if part of its vote (represented as a string of bits) encodes the solution to (say) a hard factoring problem. Of course, this is not a very satisfactory voting rule.) It should also be noted that most common voting rules are in fact monotone (on all instances), and the few rules for which nonmonotone instances can be constructed are often severely criticized because of this (even if the rule is in fact monotone on most instances).

Arguing for Property b)—most manipulable instances should be such that the manipulators can make either of exactly two candidates win—is somewhat more difficult. For simplicity, consider rules that produce a score for every candidate. As the number of voters grows, typically, candidates' scores tend to separate. This is especially the case if some candidates systematically tend to be ranked higher than others by voters, *e.g.* because these candidates are intrinsically better. (One interpretation of voting that dates back at least to Marquis de Condorcet (Condorcet 1785) is the following: everyone has a noisy signal about the relative intrinsic quality of the candidates, and the purpose of an election is to maximize the probability of choosing the intrinsically best candidate.) Thus, given a large number of nonmanipulators, it is unlikely that the scores of the two leading candidates will be close enough to each other that a few manipulators can make either one of them win; but it is significantly more unlikely that the scores of the *three* leading candidates will be close enough that the manipulators can make any one of them win. So, even given that some manipulation is possible, it is unlikely that more than two candidates can be made to win. This argument suggests that it is likely that most common voting rules in fact satisfy Property b). But it is also an argument for why we should *require* a voting rule to have this property, because, especially when we think of voting as being a process for filtering out the noise in voters' individual preferences to find the intrinsically best candidate, we *want* the candidates' scores to separate.

In the next section, we show experimentally that common voting rules in fact strongly satisfy properties a) and b).

Arguing experimentally for Property 1

In this section, we show experimentally that for all the common voting rules, most manipulable instances are in fact weakly monotone and such that either of exactly two candidates can be made to win. Because most of the rules that we study are in fact monotone on all instances, this mostly comes down to showing that at most two candidates can be made to win in most manipulable instances. Unfortunately, for quite a few of these rules, it is NP-hard to determine whether more than two candidates can win (Conitzer & Sandholm 2002; Conitzer, Lang, & Sandholm 2003). Rather than trying to solve these NP-hard problems, we will be content to provide a *lower bound* on the fraction of manipulable instances in which either of exactly two candidates can be made to win. We obtain these lower bounds by characterizing, for each rule that we study, an easily computable sufficient (but not necessary) condition for an instance to be such that either of exactly two candidates can be made to win. (We omit the definitions of the conditions due to space constraint.) For at least some rules, the lower bound is probably significantly below the actual fraction (which only strengthens the relevance of the impossibility result).

One useful property of these lower bounds is that they are independent of how the manipulators' total weight is distributed. Because of this, only the manipulators' total weight matters for the purpose of our experiments, and we can assume, without loss of generality, that each manipulator has weight 1.

For a given number of candidates, number of nonmanipulators, and number of manipulators, we generate instances as follows. We assume that there is a "correct" ranking t of the candidates (reflecting the candidates' unknown intrinsic quality), and the probability of drawing a given vote r is proportional to $p^{a(r,t)}(1-p)^{m(m-1)/2-a(r,t)}$, where $a(r,t)$ is the number of pairs of candidates on whose relative ranking r and t agree (they agree if either $c_1 \succ_r c_2$ and $c_1 \succ_t c_2$, or $c_2 \succ_r c_1$ and $c_2 \succ_t c_1$). p is a given noise parameter; if $p = 1$ then all voters will produce the correct ranking, and if $p = 0.5$ then we are drawing every vote (independently and uniformly) completely at random. This distribution is due to Condorcet (Condorcet 1785), and one way to interpret it is as follows. To draw a vote, for each pair of candidates c_1, c_2, randomly decide whether the vote is going to agree with the correct ranking on the relative ranking of c_1 and c_2 (with probability p), or disagee (with probability $1-p$). This may lead to cycles (such as $c_1 \succ c_2 \succ c_3 \succ c_1$); if so, restart. Interestingly, Young (Young 1995) observed that a voting rule proposed by Kemeny (Kemeny 1959) produces the maximum likelihood estimate of the "correct" ranking under this distribution.

These distributions often produce nonmanipulable instances. Ideally, we would discard all nonmanipulable instances, but this requires us to have an algorithm for detecting whether an instance is manipulable. If we know that the instance is weakly monotone, we can simply use algorithm Find-Two-Winners for this purpose. However, a few of the rules that we study (STV and plurality with runoff) are not monotone on all instances. In fact, for these rules, it is NP-hard to tell whether the instance is manipulable (Conitzer & Sandholm 2002; Conitzer, Lang, & Sandholm 2003). For these rules, we use simple sufficient (but not necessary) conditions to classify an instance as nonmanipulable. We will classify each nonmanipulable instance that does not satisfy this condition as having more than two potential winners, so that our results are still lower bounds on the actual ratio.

In the experiments below, we draw 1000 manipulable instances at random (by drawing and discarding instances as described above), and for each voting rule, we show our lower bound on the number of instances in which the manipulators can make either of exactly two candidates win. For rules that are not monotone everywhere, we also show a lower bound on the number of such instances that are *also* weakly monotone (indicated by "<rule> - monotone"). We also consider the Condorcet criterion, and show a lower bound on the number of instances for which these properties are satisfied for *any* rule satisfying the Condorcet criterion.

In our first experiment (Figure 1), we have three candidates, one manipulator, and significant noise in the votes ($p = 0.6$). For all the rules under study, the fraction of instances satisfying the property approaches 1 as the number of nonmanipulator votes grows.

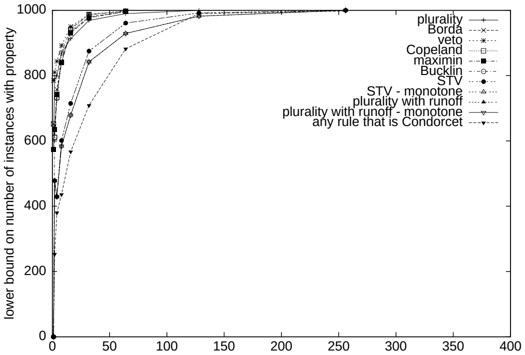

Figure 1: p = 0.6, one manipulator, three candidates.

Next, we show what happens when we maximize noise ($p = 0.5$), so that votes are drawn completely at random (Figure 2). Even under such extreme noise, the fraction of instances satisfying the property approaches 1 or at least be-

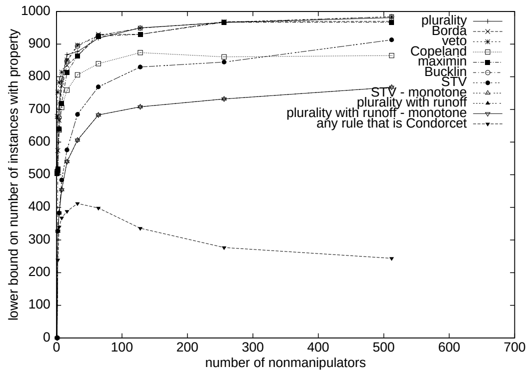

Figure 2: p = 0.5, one manipulator, three candidates.

comes very large (> 0.7) for every one of the rules. However, it is no longer possible to say this for *any* rule satisfying the Condorcet criterion (although the specific common rules that do satisfy this criterion satisfy the property for a very large fraction of instances).

Next, we show results when there are multiple (specifically, 5) manipulators (Figure 3). The results are qualitatively similar to those in Figure 1, although for smaller numbers of nonmanipulators the fractions are lower. This makes sense: when the number of nonmanipulators is relatively small, a large coalition is likely to be able to make any candidate win.

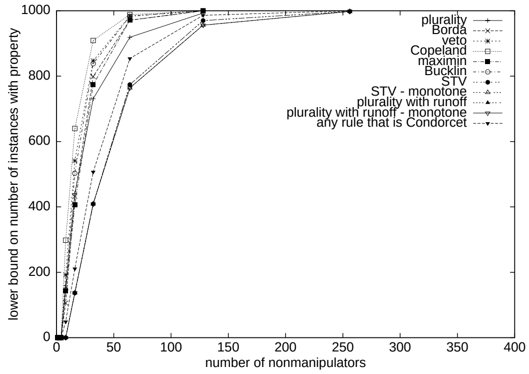

Figure 3: p = 0.6, five manipulators, three candidates.

Finally, we experiment with an increased number of candidates (Figure 4). Now, the lower bound on the fraction of instances satisfying the property approaches 1 for all rules but STV. The lower fraction for STV is probably at least in part due to the fact that the lower bound that we use for STV is relatively weak. For example, any instance in which the manipulators can change the eliminated candidate in at least two rounds is counted as having more than two candidates that the manipulators can make win. This is extremely conservative because changes in which candidate is eliminated in a given round often do not change the winner.

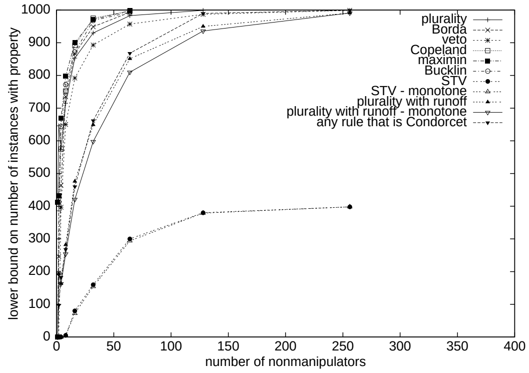

Figure 4: p = 0.6, one manipulator, five candidates.

Can the impossibility be circumvented?

One may wonder whether there are ways to circumvent the impossibility result presented in this paper. Specifically, one may still be able to construct voting rules that are usually hard to manipulate by considering a larger class of voting rules, a class that contains rules that do not satisfy the preconditions of the impossibility result. In this section, we discuss various approaches for circumventing the impossibility result, and their prospects.

Allowing low-ranked candidates to sometimes win

The impossibility result is only significant if in a sizable fraction of manipulable instances, only two candidates can be made to win. One may try to prevent this by using a voting rule that sometimes chooses as the winner a candidate that in fact did not do well in the votes (according to whatever criterion), thereby increasing the number of candidates that can be made to win in manipulable instances. Of course, having such a candidate win is inherently undesirable, but if it occurs rarely, it may be a price worth paying in order to achieve hardness of manipulation.

If we take this approach, and in addition allow for the rule to be *randomized*, then we can construct reasonable voting rules that are in fact strategy-proof (that is, no beneficial manipulation is ever possible). Consider, for example, the following voting rule:

Definition 3 *The* Copeland-proportional *rule chooses candidate c as the winner with probability proportional to c's Copeland score.*

An alternative interpretation of this rule is the following: choose a pair of candidates at random; the winner of their pairwise election wins the entire election. (If the pairwise election is tied, choose one of the two candidates at random.)

Theorem 2 *Copeland-proportional is strategy-proof.*

Gibbard has shown that a randomized voting rule is strategy-proof only if it is a probability mixture of unilateral and duple rules (Gibbard 1977). (A rule is *unilateral* if only one voter affects the outcome, and *duple* if only two candidates can win.) The Copeland-proportional rule only randomizes over duple rules (namely, pairwise elections).

Of course, the Copeland-proportional rule is still not ideal. For instance, even a Condorcet winner has a probability of only $(m-1)/(m(m-1)/2) = 2/m$ of winning under this rule. (However, this rule will at least never choose a candidate that loses every pairwise election.) Thus, it may be worthwhile to try to construct voting rules that are usually hard to manipulate, and that are more likely to choose a "good" winner than Copeland-proportional.

Expanding the definition of a voting rule

The impossibility result may cease to hold when the rule can choose from a richer outcome space. As in the previous subsection, this may prevent problems of manipulability completely, by allowing the construction of strategy-proof rules.

For example, *if* payments are possible and the agents have quasilinear utility functions, then a payment scheme such as the VCG mechanism can induce strategy-proofness. As another example that does not require these assumptions, suppose that it is possible to exclude certain voters from the effects of the election—as an illustrative example, in an election for a country's president, suppose that it is possible to *banish* certain voters to another country, in which it will no longer matter to those voters who won the election. (It does not matter whether living in the other country is otherwise more or less desirable than in the original country.) Then, we can augment any voting rule as follows:

Definition 4 *For any voting rule (in the standard sense) R, the* banishing *rule $B(R)$ always chooses the same winner as R, and banishes every pivotal voter. (A voter is* pivotal *if, given the other votes, it can make multiple candidates win.)*

Theorem 3 *For any rule R, the banishing rule $B(R)$ is strategy-proof.*

However, this scheme also has a few drawbacks. For one, it may not always be possible to completely exclude a voter from the effects of the election. Another strange property of this scheme is that no voter is ever capable of affecting its own utility, so that *any* vote is strategically optimal. Finally, it may be necessary to banish large numbers of voters. In fact, the following lemma shows that for any rule, the votes may turn out to be such that more than half of the voters must be banished.

Theorem 4 *For any responsive voting rule R, it is possible that more than half the voters are simultaneously pivotal. (We say that a voting rule is* responsive *if there are two votes r_1, r_2 such that everyone voting r_1 will produce a different winner than everyone voting r_2.)*

Rules that are hard to execute

The impossibility result only applies when an efficient algorithm is available for executing the rule, because algorithm Find-Two-Winners makes calls to such an algorithm as a subroutine. Thus, one possible way around the impossibility is to use a rule that is hard to execute. Indeed, a number of voting rules have been shown to be NP-hard to execute (Bartholdi, Tovey, & Trick 1989b; Hemaspaandra, Hemaspaandra, & Rothe 1997; Cohen, Schapire, & Singer 1999; Dwork *et al.* 2001; Ailon, Charikar, & Newman 2005). Of course, we do actually need an algorithm for executing the rule to determine the winner of the election; and, although we cannot expect this to be a worst-case polynomial-time algorithm, it should at least run reasonably fast in practice for the rule to be practical. But if the algorithm does run fast in practice, then it can also be used by the manipulators as the subroutine in Find-Two-Winners. Therefore, this approach does not look very promising.

Conclusions

Aggregating the preferences of self-interested agents is a key problem for multiagent systems, and one general method for doing so is to vote over the alternatives (candidates). Unfortunately, the Gibbard-Satterthwaite theorem shows that when there are three or more candidates, all reasonable voting rules are manipulable (in the sense that there exist situations in which a voter would benefit from reporting its preferences insincerely). To circumvent this impossibility result, recent research has investigated whether it is possible to make finding a beneficial manipulation computationally hard. This approach has had some limited success, exhibiting rules under which the problem of finding a beneficial manipulation is hard in the worst-case sense. However, this still does not preclude the existence of an efficient algorithm that *often* finds a successful manipulation (when it exists). There have been attempts to design a rule under which finding a beneficial manipulation is *usually* hard, but they have failed. To explain this failure, in this paper, we showed that it is in fact impossible to design such a rule, if the rule is also required to satisfy another property: a large fraction of the manipulable instances are both weakly monotone, and allow the manipulators to make either of exactly two candidates win. We argued why one should expect voting rules to have this property, and showed experimentally that common voting rules satisfy it. We also discussed approaches for potentially circumventing this impossibility result, some of which appear worthwhile to investigate in future research.

References

Ailon, N.; Charikar, M.; and Newman, A. 2005. Aggregating inconsistent information: Ranking and clustering. *STOC*.

Arrow, K. 1963. *Social choice and individual values*. New Haven: Cowles Foundation, 2nd edition. 1st edition 1951.

Bartholdi, III, J., and Orlin, J. 1991. Single transferable vote resists strategic voting. *Social Choice and Welfare* 8(4):341–354.

Bartholdi, III, J.; Tovey, C.; and Trick, M. 1989a. The computational difficulty of manipulating an election. *Social Choice and Welfare* 6(3):227–241.

Bartholdi, III, J.; Tovey, C.; and Trick, M. 1989b. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare* 6:157–165.

Cohen, W.; Schapire, R.; and Singer, Y. 1999. Learning to order things. *J. of Artificial Intelligence Research* 10:213–270.

Conitzer, V., and Sandholm, T. 2002. Complexity of manipulating elections with few candidates. *AAAI*, 314–319.

Conitzer, V., and Sandholm, T. 2003. Universal voting protocol tweaks to make manipulation hard. *IJCAI*, 781–788.

Conitzer, V.; Lang, J.; and Sandholm, T. 2003. How many candidates are needed to make elections hard to manipulate? *TARK*.

Davenport, A., and Kalagnanam, J. 2004. A computational study of the Kemeny rule for preference aggregation. *AAAI*, 697–702.

de Caritat (Marquis de Condorcet), M. J. A. N. 1785. Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. Paris: L'Imprimerie Royale.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the web. In *Proceedings of the 10th World Wide Web Conference*, 613–622.

Elkind, E., and Lipmaa, H. 2005a. Hybrid voting protocols and hardness of manipulation. *ISAAC*.

Elkind, E., and Lipmaa, H. 2005b. Small coalitions cannot manipulate voting. *Financial Cryptography*.

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

Gibbard, A. 1977. Manipulation of schemes that mix voting with chance. *Econometrica* 45:665–681.

Hemaspaandra, E.; Hemaspaandra, L.; and Rothe, J. 1997. Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP. *JACM* 44(6):806–825.

Kemeny, J. 1959. Mathematics without numbers. In *Daedalus*, volume 88. 571–591.

Procaccia, A. D., and Rosenschein, J. S. 2006. Junta distributions and the average-case complexity of manipulating elections. *AAMAS*.

Satterthwaite, M. 1975. Strategy-proofness and Arrow's conditions: existence and correspondence theorems for voting procedures and social welfare functions. *J. Econ. Theory* 10:187–217.

Young, P. 1995. Optimal voting rules. *J. Econ. Persp.* 9(1):51–64.