Small Coalitions Cannot Manipulate Voting

Edith Elkind[1] and Helger Lipmaa[2]

[1] Princeton University, USA
[2] Cybernetica AS and University of Tartu, Estonia

Abstract. We demonstrate how to make voting protocols resistant against manipulation by computationally bounded malicious voters, by extending the previous results of Conitzer and Sandholm in several important directions: we use one-way functions to close a security loophole that allowed voting officials to exert disproportionate influence on the outcome and show that our hardness results hold against a large fraction of manipulating voters (rather than a single voter). These improvements address important concerns in the field of secure voting systems. We also discuss the limitations of the current approach, showing that it cannot be used to achieve certain very desirable hardness criteria.

Keywords: Electronic voting, one-way functions, vote manipulation.

1 Introduction

In a democratic society, many important decisions are made based on the results of popular voting. Probably, the most frequently used voting scheme is Plurality: each voter submits a ballot with a candidate's name on it, and the candidate with the largest number of votes wins. While this is the most natural approach for the case of two candidates, when the number of candidates is larger, it does not work so well, as it is notoriously insensitive to the voters' preferences other than their top choice. Consequently, the problem of designing a more expressive voting procedure that is also fair and efficient has been a subject of extensive research.

The notions of expressiveness and efficiency are relatively easy to formalise: we assume that each voter submits a total ordering of the alternatives, best to worst (a more general definition would allow the voter to express the intensity of his preferences, but in many situations an ordering is sufficient), and, as is common in computer science, we restrict ourselves to voting schemes that given the set of votes, compute a winner in polynomial time (one should note, however, that there are interesting voting schemes that do not have this property unless $P = NP$.

On the other hand, a good definition of fairness proved to be much more elusive: in fact, the celebrated theorem of Arrow demonstrates that certain very desirable fairness criteria for a voting scheme are mutually incompatible. A system that is perceived as unfair can provoke a desire to game it (a much publicised example is provided by `http://voteswap.com`). Indeed, it has been

shown (see [Gib73, Sat73]) that any non-dictatorial voting scheme of the above-described type is susceptible to manipulation by voters, i.e., there are situations when a voter can misrepresent his preferences and achieve an outcome that he likes better than the truthful outcome. If many voters engage in these activities, the outcome of the election may be seriously distorted, and the results will not be representative of the true distribution of preferences. Thus, vulnerability to manipulation is a serious problem that has to be addressed.

While from information-theoretic perspective no solution is possible, computational considerations come to rescue: one can try to discourage potential manipulators by making manipulation infeasible. Indeed, it is known [BO91] that certain voting schemes (e.g., Single Transferable Vote) are NP-hard to manipulate. Furthermore, in a recent sequence of papers [CS02, CS03], Conitzer and Sandholm showed that several other voting schemes can be uniformly modified so that manipulating them becomes computationally hard. This is achieved by adding a pre-round in which candidates are divided into pairs and the voters' preferences are used to determine the winner of each pair; the winners of this stage participate in elections conducted according to the original protocol. Different methods for pairing up the candidates and eliciting the votes give rise to different levels of complexity, such as NP-hardness, #P-hardness, or PSPACE-hardness.

Conitzer and Sandholm leave it as an open question whether this approach can make manipulation as hard as inverting one-way functions, with the main motivation of achieving some kind of an average-case hardness. This question is particularly interesting because the setting of voting manipulation is reminiscent of that of common cryptographic tasks, where the goal is to construct a protocol that can withstand an attack by malicious adversary with overwhelming probability, i.e., on average, assuming that the adversary computationally bounded.

Motivated by this question of Conitzer and Sandholm, we modify their construction so that the pre-round schedule is computed from the votes of all voters using a one-way function. Since the size of the argument to this function is determined by the number of candidates m, our results are interesting only if m is large (this is also the case in [CS02, CS03]): for all voting schemes considered in this paper, manipulation is easy when the number of candidates m is very small, and in Section 4, we show that in some sense, this is inevitable. However, in some real-life scenarios (most notably, recent California gubernatorial elections), the number of candidates is sufficient to make our results applicable.

Unfortunately, we do not achieve average-case hardness (some reasons why this is unlikely to be possible are outlined in Section 4). The good news is that by using a deterministic one-way function to fix the scheduling we achieve some other attractive properties.

First, we show that our method can be used to make voting protocols hard to manipulate even by a large minority fraction (1/6 in the case of Plurality, STV and Maximin, $1/m$ in the case of Borda) of voters, while the previous literature concentrated on the case of a single manipulator. Arguably, the voters who want

to manipulate the election may and do collude, so constructing voting schemes that are secure against a significant fraction of cheaters is an important goal.

Second, the paper [CS03] assumes that the election officials can be trusted with constructing the pre-round schedule, which is supposed to be generated at random for NP-hardness and #P-hardness results (before and after vote elicitation, respectively). Forcing the election authorities to prove that they, indeed, used a secure random number generator rather than paired up the candidates at their will is hard to achieve in practice. On the other hand, it is clear that in many cases malicious pre-round scheduling can be used to eliminate an "undesirable" candidate or affect the election results in some other way, so if the entities responsible for scheduling are corrupted, there is a huge incentive for them to deviate from the protocol. Our approach addresses this issue by extracting the necessary randomness from the votes themselves (for a more rigorous description, see Section 3); this limits the potential for cheating by (possibly corrupt) officials. Moreover, the voters do not need to rely on any external randomness for their own actions either, since the voting scheme is completely deterministic.

The rest of the paper is organised as follows. In Section 2 we introduce our notation, give a precise definition of what it means to manipulate an election, and describe some well-known voting schemes that can be made secure using our approach. For completeness, we also provide the definition of one-way functions, and state some related facts. In Section 3, we describe our constructions for specific protocols. In Section 4, we discuss the limitations of this approach to making manipulation hard. Section 5 presents our conclusions and future research directions.

2 Preliminaries and Notation

We assume that there are n voters and m candidates and denote the set of all voters by $V = \{v_1, \ldots, v_n\}$ and the set of all candidates by $C = \{c_1, \ldots, c_m\}$. Our complexity results are in terms of m and n, i.e., unless specified otherwise, 'polynomial' always means 'polynomial in m and n'.

The set of all permutations of C is denoted by $\Pi(C)$; a voter j's preferences are expressed by a list $\pi_i \in \Pi(C)$: the first element is the voter's most preferred candidate, etc. In particular, this means that within one voter's preference list, ties are not allowed. A *voting scheme* is a mapping $P : \langle \Pi(C), \ldots, \Pi(C) \rangle \mapsto C$ that selects a winner $c \in C$ based on all voters' preference lists.

To state our results formally, we need to define more precisely what we mean by beneficial manipulation. We distinguish between *constructive manipulation*, which is a misrepresentation of a voter's preferences that makes his top candidate an overall winner, and *destructive manipulation*, i.e., an untruthful vote that replaces the actual winner (according to the true preferences) with a candidate that the manipulator prefers over the actual winner; clearly, the second notion is strictly weaker than the first one.

We say that a voter v_j can *manipulate* a protocol P if he can find a permutation $\pi'_j \in \Pi(C)$ such that for some values of $\pi_i \in \Pi(C)$, $i = 1, \ldots, n$, we have

1. $P(\pi_1, \ldots, \pi_n) = c$;
2. $P(\pi_1, \ldots, \pi_{j-1}, \pi'_j, \pi_{j+1}, \ldots, \pi_n) = c' \neq c$;
3. v_j ranks c' above c.

We say that v_j manipulates P *constructively* if v_j ranks c' first and *destructively* otherwise; v_j manipulates P *efficiently* if there is a probabilistic polynomial time algorithm that given preference lists $\pi_1, \ldots, \pi_n$ for which such π'_j exists, constructs π'_j with non-negligible probability (over the coin tosses of the algorithm).

We say that a set of voters M with $\tau = |M|$ can $(\tau, n = n(\tau))$-*manipulate* a protocol P if there is a pair of all voters' preference profiles $(\pi = (\pi_1, \ldots, \pi_n), \pi' = (\pi'_1, \ldots, \pi'_n))$, such that $\pi_i = \pi'_i$ for $i \notin M$, and everyone in M strictly prefers the outcome of P on π' to the outcome of P on π. The manipulation is *constructive* if everyone in M ranks $P(\pi')$ first, and *efficient* if whenever such π' exists, it can be constructed by a probabilistic polynomial time algorithm with non-negligible probability.

Common Voting Protocols. In this paper, we consider the following common voting protocols (in all these definitions, the candidate with the most points wins):

- *Plurality.* A candidate receives 1 point for every voter that ranks it first.
- *Borda.* For each voter, a candidate receives $m - 1$ points if it is the voter's top choice, $m - 2$ if it is the second choice, $\ldots$, 0 if it is the last.
- *Single Transferable Vote (STV).* The winner determination process proceeds in rounds. In each round, a candidate's score is the number of voters that rank it highest among the remaining candidates, and the candidate with the lowest score drops out. The last remaining candidate wins. (A vote transfers from its top remaining candidate to the next highest remaining candidate when the former drops out.)
- *Maximin.* A candidate's score in a pairwise election is the number of voters that prefer it over the opponent. A candidate's number of points is the lowest score it gets in any pairwise election.

Pre-Round. We reproduce the definition of *pre-round* [CS03] for reader's convenience:

1. The candidates are paired. If there is an odd number of candidates, one candidate gets a bye.
2. In each pairing of two candidates, the candidate losing the pairwise election between the two is eliminated. A candidate with a bye is never eliminated.
3. On the remaining candidates, the original protocol is executed to produce the winner. For this, the implicit votes over the remaining candidates are used.

The schedule of the pre-round is an ordering S_m of m candidates (it is assumed that in the pre-round, candidate $S_m(2i - 1)$ is matched with $S_m(2i)$, $i = 1, \ldots, \lfloor m/2 \rfloor$, and if m is odd, $S_m(m)$ gets a bye). We denote the protocol that consists of a base protocol P (such as Plurality or Borda) preceded by a pre-round that is scheduled according to S_m by $S_m - P$.

One-Way Functions. A function $f : \{0,1\}^* \mapsto \{0,1\}^*$ is *one-way* if

- There exists a probabilistic polynomial-time algorithm (PPT) that on input x outputs $f(x)$;
- For every PPT A and every polynomial $p(k)$, for sufficiently large k it holds that
$$P\left[f(z) = f(x) : x \xleftarrow{R} \{0,1\}^k; z \leftarrow A(1^k, f(x))\right] \leq \frac{1}{p(k)} \ .$$

Note that to prove that a function f is *not* one-way it suffices to exhibit an infinite sequence $k_1, k_2, \ldots$ and an efficient algorithm A that inverts f on inputs of length k_i. It is well-known that any one-way function can be transformed into a *length-preserving* one-way function, i.e., one that maps inputs of length k to outputs of length k. Hence, assuming that one-way functions exist is equivalent to assuming that length-preserving one-way functions exist.

3   Reduction Based on One-Way Functions

Here we show that if one-way functions exist, then for several protocols adding a pre-round with a carefully constructed schedule makes constructive manipulation hard. We consider a family of pre-round schedules parameterised by a pair of functions (k, f), where $k : \mathbb{N} \mapsto \mathbb{N}$ is any function that satisfies $k(m) < \log_2(\lfloor m/2 \rfloor!)$ and $f : \{0,1\}^* \mapsto \{0,1\}^*$ is a length-preserving function; an element of this family is denoted by $S_m^{k,f}$.

We demonstrate that if f is a length-preserving one-way function and $k(m)$ is chosen in a certain way (which might be different for different base protocols), then manipulating $S_m^{k,f} - P$ is as hard as inverting f. In what follows, we describe the $S_m^{k,f}$ used in our construction in full detail.

Definition of $S_m^{k,f}$. As in [CS03], we define a match-up slot to be a space in the pre-round in which two candidates can face each other. Fix m and set $k = k(m)$. Let t be the smallest integer that satisfies $t! > 2^k$. Choose $2t$ candidates arbitrarily; assign t of them to the first t match-up slots. Pair up the remaining $m - 2t$ candidates; if there is an odd number of them, one candidate gets a bye. Assign them to the remaining slots. Now all that has to be chosen is the match-up slots for the t unscheduled candidates. Renumber them from 1 to t. Elicit the votes.

For each voter v_i, $i = 1, \ldots, n$, compute a string $s_i \in \{0,1\}^k$ as follows. Suppose that v_i orders the unscheduled candidates as $(c_1^i, \ldots, c_t^i)$. Find the lexicographic number of $(c_1^i, \ldots, c_t^i)$ in the set of all permutations over $\{1, \ldots, t\}$ and let s_i be the last k digits in the binary representation of this number. Note that since $t! > 2^k$, every string of length k can be obtained in this way. Let $s = \oplus_{i=1}^n s_i$. Compute $f(s)$, and denote the permutation whose lexicographic number is $f(s)$ by $(c_{i_1}, \ldots, c_{i_t})$ (again, the existence of such permutation is guaranteed since $t! > 2^k$). Assign the c_{i_1}th candidate to the first slot, c_{i_2}nd

candidate to the second slot, etc. This method of pairing up the candidates implicitly describes $S_m^{k,f}$.

Our reduction is based on the following idea: we choose the preferences of non-manipulating voters so that the actual vote of the manipulators does not affect the election results apart from its use for pre-round scheduling. Furthermore, given the votes of others, there is only one way to schedule the candidates in the pre-round so that the preferred candidate will win. Hence, to achieve their goal, the manipulators must guess the pre-image of the desired permutation under the chosen one-way function.

Note that for some protocols (namely, Plurality and STV) it is possible to set $t = \lfloor m/2 \rfloor$. In this case, our pre-round scheduling method has a very natural interpretation: we separate the candidate pool into two approximately equal components, and stipulate that two candidates from the same component should not compete with each other in the pre-round. Furthermore, candidates from different components are matched randomly, where randomness is derived from the input itself (i.e., the votes), rather than from an external source.

Results and Proofs for Specific Protocols. In this subsection, we give three rather similar proofs for pre-round versions of Plurality, STV, and Maximin voting protocols. All of them are proven to be secure against approximately 1/6 of manipulators.

Theorem 1. *Assume that m is even, and let $t = m/2$ and $k = \lfloor \log_2(t!) \rfloor$ (for $k \geq 80$ it must be that $t \geq 25$ and thus $m \geq 52$). Then there is a polynomial-time algorithm that can invert f on inputs of length k using an oracle that can constructively $(\tau, 6\tau + 5)$-manipulate $S_m^{k,f} -$ Plurality.*

Corollary 1. *If one-way functions exist, there is a pair of functions $f : \{0,1\}^* \mapsto \{0,1\}^*$ and $k : \mathbb{N} \mapsto \mathbb{N}$ such that no polynomial-time adversary can constructively $(\tau, 6\tau + 5)$-manipulate $S_m^{k,f} -$ Plurality for infinitely many values of m.* $\qquad\square$

Similar corollaries can be derived from other theorems in this section in a straightforward manner; we will not state them explicitly.

Proof (of Thm. 1). We show how to invert f on a random input using an algorithm that allows τ voters to find a constructive manipulation of the protocol for m candidates and $6\tau + 5$ voters whenever one exists, and carefully constructed preference lists for the $5\tau + 5$ non-manipulators. That is, we describe an algorithm that when given $Y = f(X)$, where X is chosen uniformly at random from $\{0,1\}^k$, finds a Z such that $f(Z) = Y$ with non-negligible probability.

First, find a permutation $(a_1, \ldots, a_t)$ whose lexicographic number is Y. Let the m candidates be

$$x_1, y_1, x_2, y_2, \ldots, x_{t-1}, y_{t-1}, p, z,$$

and let each of the τ manipulators prefer p to any other candidate. Assign $x_{a_1}, \ldots, x_{a_{t-1}}, p$ to the first t match-up slots. Set the non-manipulator votes as follows:

$$x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > p > z$$
$$\text{— } 2(\tau + 1) \text{ votes}$$

$$y_{a_1} > y_{a_2} > \cdots > y_{a_{t-1}} > p > x_{a_1} > \cdots > x_{a_{t-1}} > z$$
$$\text{— } 2(\tau + 1) \text{ votes}$$

$$p > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > z$$
$$\text{— } \tau + 1 \text{ votes.}$$

We observe the following:

(1) In the pairwise election, y_{a_1} can only be eliminated by x_{a_1}, y_{a_2} can only be eliminated by x_{a_1}, y_{a_1}, and x_{a_2}, etc., so to eliminate all y_is, x_{a_j} has to be scheduled with y_{a_j} for $j = 1, \ldots, t-1$, and p has to be scheduled with z.

(2) If all y_i are eliminated in the pre-round, p gets at least $2\tau + 2 + \tau + 1$ votes, i.e., a majority, and wins.

(3) Suppose that some of the y_i survive the pre-round. Then y_{a_i} with the smallest i among them gets at least $2\tau + 2$ votes, while p gets at most $\tau + 1 + \tau = 2\tau + 1$ votes. Hence, in this case p does not win.

Now, the rest of the reduction is straightforward. We have seen that the manipulators' vote only affects the outcome by being used for pre-round scheduling, and furthermore, p only wins if all y_i's are eliminated in the pre-round. Hence, to get p to win, the manipulators need to order $y_1, \ldots, y_{t-1}, z$ in their votes so that the resulting s_i's (and, consequently, s) are such that when $f(s)$ is used for pre-round scheduling, y_{a_1} gets assigned to the 1st slot, y_{a_2} gets assigned to the 2nd slot, etc. By observing their votes, we can compute s_i; since all other votes are publicly known, we can also compute s_j, $j \neq i$, and, consequently, s. This s satisfies $f(s) = Y$, so we have found a pre-image of Y under f. $\qquad \square$

Theorem 2. *Assume that m is even and let $t = m/2$ and $k = \lfloor \log_2(t!) \rfloor$. Then there is a polynomial-time algorithm that can invert f on inputs of length k using an oracle that can constructively $(\tau, 6\tau + 5)$-manipulate $S_m^{k,f} - \mathsf{STV}$.*

Proof. Again, we are given $Y = f(X)$, where X is chosen uniformly at random from $\{0,1\}^k$, and want to find a Z such that $f(Z) = Y$ with non-negligible probability. We start by finding a permutation $(a_1, \ldots, a_t)$ whose lexicographic number is Y. Let the set of candidates be

$$x_1, y_1, x_2, y_2, \ldots, x_{t-1}, y_{t-1}, p, z,$$

and let each of the τ manipulators prefer p to any other candidate.

Assign $x_{a_1}, \ldots, x_{a_{t-1}}$ and p to the first t match-up slots. Set the non-manipulator votes as follows:

$$z > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > p$$
$$\text{— } 2(\tau + 1) \text{ votes}$$

$$y_{a_1} > y_{a_2} > \cdots > y_{a_{t-1}} > p > z > x_{a_1} > x_{a_2} > \ldots > x_{a_{t-1}}$$
$$\text{— } 2(\tau + 1) \text{ votes}$$

$$p > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_{t-1}} > y_{a_{t-1}} > z$$
$$\text{— } \tau + 1 \text{ votes.}$$

We observe the following:

(1) In the pairwise election, p is preferred over z, but not over any of the y_i, so, to survive the pre-round, p has to be scheduled with z.
(2) In the pairwise election, y_{a_1} can only be eliminated by x_{a_1}, y_{a_2} can only be eliminated by x_{a_1}, y_{a_1}, and x_{a_2}, etc., so to eliminate all y_i, we have to schedule x_{a_j} with y_{a_j} for $j = 1, \ldots, t - 1$.
(3) If all y_i are eliminated in the pre-round, in the beginning of the main round p has more than a half of the votes, so he wins.
(4) Suppose that some of the y_i survive the pre-round. Then after τ rounds of STV the first $2\tau + 2$ votes go to either z or x_1, the highest ranking of the surviving y_i gets $2\tau + 2$ votes as well, and p gets at most $2\tau + 1$ votes. Hence, at this point p will be eliminated, so he does not win the election.

The rest of the argument is as in the previous case. The total number of voters is $n = 6\tau + 5$. □

Theorem 3. *Suppose that m is even and let $t = \lfloor m/2 \rfloor - 4$ and $k = \lfloor \log_2(t!) \rfloor$. Then there is a polynomial-time algorithm that can invert f on inputs of length k using an oracle that can constructively $(\tau, 6\tau + 5)$-manipulate $S_m^{k,f} - \mathsf{Maximin}$.*

Proof. Again, we are given $Y = f(X)$, where X is chosen uniformly at random from $\{0, 1\}^k$, and want to find a Z such that $f(Z) = Y$ with non-negligible probability. We start by finding a permutation $(a_1, \ldots, a_t)$ whose lexicographic number is Y. Let the set of candidates be

$$x_1, y_1, x_2, y_2, \ldots, x_t, y_t, p, z_1, z_2, z_3,$$

and let each of the τ manipulators prefer p to any other candidate.

Assign $x_{a_1}, \ldots, x_{a_t}$ to the first t match-up slots. Pair up p, z_1, z_2, and z_3, and assign them to the last two slots. Set the non-manipulator votes as follows:

$$x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_t} > y_{a_t} > p > z_1 > z_2 > z_3$$
$$\text{— } 2(\tau + 1) \text{ votes}$$

$$z_1 > z_2 > z_3 > y_{a_1} > y_{a_2} > \cdots > y_{a_t} > p > x_{a_1} > x_{a_2} > \ldots > x_{a_t}$$
$$\text{— } 2(\tau + 1) \text{ votes}$$

$$p > z_1 > z_2 > z_3 > x_{a_1} > y_{a_1} > x_{a_2} > y_{a_2} > \cdots > x_{a_t} > y_{a_t}$$
$$\text{— } \tau + 1 \text{ votes.}$$

We observe the following:

(1) Both p and exactly one of the z_i, which we denote by z, survive the pre-round.
(2) In the pairwise election, y_{a_1} can only be eliminated by x_{a_1}, y_{a_2} can only be eliminated by x_{a_1}, y_{a_1}, and x_{a_2}, etc., so to eliminate all y_i, we have to schedule x_{a_j} with y_{a_j} for $j = 1, \ldots, t$.
(3) Suppose that any of the y_i survives the pre-round. Then p's score is at most $2\tau + 1$ (there are only $\tau + 1$ honest voters that prefer it over any of the remaining y_i), and z's score is at least $2\tau + 2$ (there are $2\tau + 2$ honest voters that rank it first), so p cannot win the elections.
(4) Suppose that none of the y_i survives the pre-round. Then p's score is at least $3\tau + 3$ (there are $3\tau + 3$ honest voters that prefer it over any of the x_i and $3\tau + 3$ honest voters that prefer it over z), z's score is at most $3\tau + 2$ (there are only $2\tau + 2$ honest voters that prefer it over p), and the score of any of the x_i is at most $3\tau + 2$, since there are only $2\tau + 2$ honest voters that prefer it over z. Hence, in this case p wins.

The rest of the argument is as in the previous case. The total number of voters is $n = 6\tau + 5$.   □

While the previous results guaranteed security against a constant fraction of manipulators, in the case of the Borda protocol the allowable fraction of manipulators depends on the total number of candidates.

Theorem 4. *Let $m > 4$ be the number of candidates and n be the number of voters. Suppose that f is a one-way function, $m - 4t^2 - 8t + 1 > 0$, $m = \mathrm{poly}(t)$, and $k = \lfloor \log_2(t!) \rfloor$ (for $k \geq 80$ it must be that $t \geq 25$ and thus $m \geq 1300$). Then there is a polynomial-time algorithm that can invert f using an oracle that can find a constructive $(\tau, (m + 4t + 8)\tau)$-manipulation of $S_m^{k,f} - \mathsf{Borda}$ whenever one exists.*

Proof. For Borda protocol, set $d = m - 2t - 1$, and let the set of candidates be

$$x_1, y_1, \ldots, x_t, y_t, p, z_1, \ldots, z_d,$$

and there are τ manipulators who rank p first.

Assign $x_{a_1}, \ldots, x_{a_t}$ to the first match-up slots. Pair up $p, z_1, \ldots, z_d$ and assign them to the remaining slots. Set the non-manipulator votes as follows:

$$x_{a_1} > y_{a_1} > \ldots > x_{a_t} > y_{a_t} > p > z_1 > \ldots > z_d \text{ --- } \alpha \text{ votes,}$$
$$p > y_{a_1} > \ldots > y_{a_t} > z_1 > \ldots > z_d > x_{a_1} > \ldots > x_{a_t} \text{ --- } \beta \text{ votes,}$$

where α and β are to be determined later. Assume for convenience that m is even. Discarding the votes of τ manipulators, we observe the following:

(1) The preferred candidate p survives the pre-round, and in the main round, p gets more points than any of the surviving z_i.

(2) If all y_i are eliminated in the pre-round (and hence all x_i survive), x_{a_1} gets $\left(\frac{m}{2} - 1\right)\alpha + (t-1)\beta$ points and other x_i get fewer points, while p gets $\left(\frac{m}{2} - t - 1\right)\alpha + \left(\frac{m}{2} - 1\right)\beta$ points. Thus p wins as soon as $\left(\frac{m}{2} - t\right)\beta - t\alpha > 0$.

(3) If any of the y_i is not eliminated in the pre-round, then in the main round the highest-ranking of the surviving y_i, which we denote by y_{i_0}, gets at least $\alpha - \beta$ more points than p (the first α votes rank y_{i_0} higher than p, and in the last β votes, after we delete all y_i that were eliminated in the pre-round, y_{i_0} immediately follows p). Hence, if $\alpha - \beta > 0$, p cannot win.

(4) In the pairwise election, as long as $\alpha - \beta > 0$, y_{a_1} can only be eliminated by x_{a_1}, y_{a_2} can only be eliminated by x_{a_1}, y_{a_1} and x_{a_2}, etc, so to eliminate all y_i we have to schedule x_{a_i} with y_{a_i} for $i = 1, \ldots, t$.

Moreover, if we replace 0 with $(m-1)\tau$ in the right-hand side of the inequalities in (2)–(4), then these properties hold even if we add τ additional votes.

Hence, we would like to choose α and β so that $(m/2 - t)\beta - t\alpha > (m-1)\tau$, $\alpha - \beta > (m-1)\tau$, and $\alpha + \beta$ is as small as possible. From the second condition, it is clear that $\alpha + \beta = \Omega(m\tau)$; choosing $\alpha = (m + 2t + 4)\tau$, $\beta = (2t + 4)\tau$ matches this lower bound and satisfies both conditions provided that $m > 4t^2 + 8t - 1$.

$\square$

4 Limitations

The examples constructed in the previous section (as well as the ones in [CS03]) show that it is possible to construct a voting scheme that does not allow for a universal polynomial-time algorithm for finding a beneficial manipulation (under standard assumptions, such as $P \neq NP$).

However, this does not exclude the possibility that in many contexts the manipulator can figure out what to do. It would be desirable to have a voting scheme with the following property: for any voter and any vector of other voters' preference profiles finding an action that is always no worse and sometimes better than truthfully reporting your preferences is hard. The appropriate notion of hardness would be some flavour of hardness on average, such as inverting one-way functions. Moreover, we can relax this criterion by requiring it to hold with an overwhelming probability over the honest voters' preference profiles rather than all preference profiles (note, however, that to formalise this, we need to know the distribution of the voters' preferences).

Unfortunately, it turns out that this goal is impossible to achieve by simply adding a pre-round. To formally show this, we construct an example in which a (destructive) manipulation is always easy to find. Namely, we demonstrate a family of preferences profiles such that

– if everyone votes honestly, then under any pre-round schedule candidate p survives the pre-round and goes on to win the elections;
– there is a manipulation by a single voter such that for any pre-round schedule the result of the elections is a draw between p and some other candidate (and the manipulator prefers this candidate to p).

Our example is constructed for Plurality protocol, but it is possible to construct similar examples for other protocols as well.

Suppose that $m \geq 8$ is even, and set $t = m/2 - 2$. Let the set of candidates be $p, a, b, c_1, c_2, \ldots, c_{2t+1}$. Choose an arbitrary k so that $n/3 + 1 < k < n/2$. Suppose that the honest voters can be divided into three groups

- $k + 1$ honest voters whose votes are of the form

$$p > a > b > c_{j,i_1} > \cdots > c_{j,i_{2t+1}},$$

  where $j = 1, \ldots, k + 1$, and each $(c_{j,i_1}, \ldots, c_{j,i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$;
- k honest voters whose votes are of the form

$$a > b > p > c_{j,i_1} > \cdots > c_{j,i_{2t+1}},$$

  where $j = 1, \ldots, k$, and each $(c_{j,i_1}, \ldots, c_{j,i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$;
- $n - 2k - 2$ honest voters whose votes are of the form

$$c_{j,i_1} > \cdots > c_{j,i_{2t+1}} > p > a > b,$$

  where $j = 1, \ldots, n - 2k - 2$, and each $(c_{j,i_1}, \ldots, c_{j,i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$.

Suppose also that the manipulator's (honest) preference list is

$$c_{i_1} > \cdots > c_{i_{2t+1}} > a > b > p,$$

where $(c_{i_1}, \ldots, c_{i_{2t+1}})$ is a permutation of $c_1, \ldots, c_{2t+1}$.

Observe the following:

1. No matter how the manipulator votes, p always survives the pre-round.
2. If either of a or b is not matched with p in the pre-round, he survives the pre-round, so at least one of them participates in the main round.
3. At least one of c_i survives the pre-round.

Hence, if everyone votes honestly, after the pre-round there will be $k+1$ votes for p, k votes for a or k votes for b, and at most $n - 2k - 1 < k$ votes for any of the c_i. However, if the manipulator puts a and b on the top of his list, i.e., votes

$$a > b > c_1 > \cdots > c_{2m+1} > p,$$

he can achieve a draw between a/b and p. Unless the draws are always resolved so that p wins, this strictly improves the outcome of the elections from the manipulator's point of view, and in the latter case we can modify the example by increasing the number of honest voters who rank a first and b second to $k+1$, in which case the manipulator can change the situation from a draw between p and another candidate to a win by another candidate.

While under uniform distribution of preferences, the likelihood of this type of profile is not very high, the uniformity assumption itself is hardly applicable to real-life scenarios. In a more polarised society, this preference profile has a natural interpretation: suppose that there are three established parties, two of which have similar positions on many issues, and a multitude of independent candidates, and the voters can be divided into two groups: traditionalists, who do not trust any of the independent candidates, and protesters, who rank the established candidates after the independent candidates. Under some additional assumptions, the situation described above becomes quite likely.

Manipulation for Small Number of Candidates. Clearly, when the number of candidates m is constant, and there is only one manipulator, he can easily figure out what to do simply by checking the outcome for all $m!$ possible votes and submitting the one that produces the best possible result. When the number of manipulators is large, as is the case in our scenario, enumerating the space of all possible votes by the manipulating coalition becomes infeasible (even if we assume that all voters are treated symmetrically, the size of this space is still exponential in the coalition size), so it might still be the case that choosing the best possible action is hard. However, if the manipulators' goal is simply to submit a set of votes that results in a specific pre-round schedule, and the scheduling algorithm treats all candidates symmetrically, then, since the number of possible pre-round schedules is constant, this goal is likely to be attained by random guessing. Therefore, making manipulation infeasible when the number of candidates is small cannot be achieved by this method.

5   Conclusions and Future Research

Our work extends the results of [CS02, CS03] in several important directions. All our improvements address important concerns in the field of secure voting systems. First, we show that our hardness results hold against a large fraction of manipulating voters (rather than a single voter). Also, while the original protocol of [CS03] makes it possible for dishonest election authorities to affect the results by constructing the pre-round schedule in a way that suits their goals (rather than randomly), we eliminate this loophole by making the schedule dependent on the contents of the voters' ballots. Finally, voters do not need to trust any external randomness since their voting procedure is completely deterministic; in a certain sense, our pre-round construction extracts randomness from the votes.

It is important to note that our methodology, as well as the one of [CS03] works for a wide range of protocols: while some voting procedures are inherently hard to manipulate, they may not reflect the decision-making procedures that are acceptable in a given culture, and may thus be deemed inappropriate. On the other hand, a pre-round followed by an execution of the original protocol retains many of the desirable properties of the latter. All of the voting protocols described in Section 2, as well as many others, are used in different contexts;

it is unreasonable to expect that all of them will be replaced, say, by STV just because it is harder to manipulate.

Note also that while our results have been worded in the terms of polynomial time, it is relatively simple to estimate the tightness of the reductions. This is since in all four cases, the attacker that inverts f invites the $S_m^{k,f} -$ Mechanism-breaking oracle only once.

In Section 4, we discuss the limitations of the current approach, showing that it cannot be used to achieve certain very desirable hardness criteria. We leave it as an open problem whether there are other methods that satisfy these criteria, or whether there is a less ambitious set of desiderata that is acceptable in practice. Another question relates to the fraction of manipulators against which our system is secure: it would be interesting to raise this threshold from $1/6$th fraction of the voters for Plurality, STV, and Maximin, and $1/m$th for Borda, or to show that this is impossible.

References

[BO91]   John J. Bartholdi, III and James B. Orlin. Single Transferable Vote Resists Strategic Voting. *Social Choice and Welfare*, 8(4):341–354, 1991.

[CS02]   Vincent Conitzer and Tuomas Sandholm. Complexity of Manipulating Elections with Few Candidates. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence*, pages 314–319, Edmonton, Alberta, Canada, July 28 — August 1 2002. AAAI Press.

[CS03]   Vincent Conitzer and Tuomas Sandholm. Universal Voting Protocol Tweaks to Make Manipulation Hard. In *In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 781–788, Acapulco, Mexico, August 9–15 2003.

[Gib73] Allan F. Gibbard. Manipulation of Voting Schemes: A General Result. *Econometrica*, 41:597–601, 1973.

[Sat73] Mark A. Satterthwaite. *The Existence of Strategy-Proof Voting Procedures: A Topic in Social Choice Theory*. PhD thesis, University of Wisconsin, Madison, 1973.