On The Complexity of Manipulating Elections

Tom Coleman **Vanessa Teague**

Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010
Email: {colemant, vteague}@csse.unimelb.edu.au

Abstract

We study the *manipulation* of voting schemes, where a voter lies about their preferences in the hope of improving the election's outcome. All voting schemes are potentially manipulable. However, some, such as the Single Transferable Vote (STV) scheme used in Australian elections, are resistant to manipulation because it is $\mathcal{NP}$-hard to compute the manipulating vote(s). We concentrate on STV and some natural generalisations of it called Scoring Elimination Protocols. We show that the hardness result for STV is true only if both the number of voters and the number of candidates are unbounded—we provide algorithms for a manipulation if either of these is fixed. This means that manipulation would not be hard in practice when either number is small. Next we show that the weighted version of the manipulation problem is $\mathcal{NP}$-hard for all Scoring Elimination Protocols except one, which we provide an algorithm for manipulating. Finally we experimentally test a heuristic for solving the manipulation problem and conclude that it would not usually be effective.

1 Introduction

In the 2004 federal election, Family First candidate Steve Fielding won the sixth Victorian senate seat with only 45,260 first-preference votes, about 2%. It is common for the sixth seat to be won almost entirely on the basis of passed preferences, but this case was unusual because the preferences he received were from Green, Democrat and Labor voters, who probably preferred each other to the Family First party (Colebatch 2004). So why did those people (or the people who determined their senate tickets) vote that way? Perhaps they were attempting to *manipulate* the voting system, deliberately making a vote that differed from their true preferences in the hope of producing a better outcome than the truth would have provided. Obviously, if everyone lies about their preferences then the outcome may be quite unpopular. In this case, the attempt to manipulate the system failed, producing an outcome that (presumably) almost none of the manipulators expected.

Voting also arises in agent-based systems, where we face the problem of how a group of selfish agents can agree to (temporarily) cooperate on a particular course of action. Each agent may have a different set of preferences over possible actions. The system designer must choose a *voting scheme* which, given a set of *candidate* actions and a set of *votes* ordering those candidates, chooses a single winning candidate.

In agent-based systems as in the Senate, we would like to use a system that was impossible to manipulate. However, it was famously shown (Gibbard 1973, Satterthwaite 1975) that **all** non-dictatorial[1] voting protocols of at least 3 candidates are manipulable. Fortunately, all is not lost. Bartholdi and Orlin (1991) showed that it is $\mathcal{NP}$-hard to compute a successful manipulation of the STV protocol. This $\mathcal{NP}$-hardness means that it is *difficult* to manipulate such elections (especially from an agent based perspective).

Because it is known to be resistant to manipulation, we concentrate on STV and some generalisations in this paper. The STV protocol (also known as Preferential Voting, or Instant Runoff Voting) proceeds in rounds, in each of which we eliminate a single candidate. We begin by distributing votes to candidates based on first preference, and we eliminate the candidate with least votes. After any candidate is eliminated, we redistribute each of his votes to the next (remaining) preference. We re-tally the votes, and eliminate the candidate with the next fewest votes. When one candidate remains, we proclaim her the winner. STV is used in many different political electorates, notably both upper and lower house voting in most states of Australia. STV is non-dictatorial, so of course (by Gibbard and Satterthwaite) is manipulable for just three candidates.

As a simple example where manipulation of STV is possible, suppose you are the last voter in an election and the others have voted as follows over three candidates $\{a, b, c\}$:

6 votes	2 votes	2 votes	4 votes
a	b	b	c
b	a	c	b
c	c	a	a

Your true preference is $a \succ b \succ c$. If you vote that way, a will have 7 first preference votes, whilst b and c will both have 4. One of b and c will be eliminated next—if it is c, then all its votes will be redistributed to b and b will now have 8 votes, beating your favoured candidate a. So it would be better to vote $c \succ a \succ b$—if you can guarantee the elimination of b first, then after redistribution a will have a total of 8 votes, whilst c will have just 7.

In this paper we concentrate on hardness results for a new generalisation of the STV protocol, which we base on the idea of a *scoring protocol*—one in which each preference is worth a certain number of points to each voter. Hemaspaandra and Hemaspaandra (2005) show that all scoring protocols bar one are $\mathcal{NP}$-hard to manipulate when voters have different weights. In

[1] i.e., the outcome not determined by a single voter.

this paper we demonstrate some simple conditions under which it is easy to manipulate such protocols.

Elkind and Lipmaa (2005) demonstrate a method of taking different protocols and combining them through an operation called *hybridization*. They accomplish this by using an elimination step in the first protocol. They consider the idea of closing a protocol under hybridization with itself, forming a new protocol that is significantly different, but in some ways the same. We extend this idea, and consider the closure of scoring protocol, termed a *scoring elimination protocol*—these form a class of protocols generalising `STV`.

We know that `STV` is $\mathcal{NP}$-hard to manipulate. However, this fact relies on instances with large numbers of both candidates and voters. Conitzer and Sandholm (2002) demonstrate that for all $\mathcal{P}$-time evaluable protocols, manipulation is in $\mathcal{P}$ when the number of candidates is fixed. However this algorithm has extremely high complexity, and thus is impractical for any but very small numbers of candidates. In this paper we provide a more efficient algorithm for `STV`. We also provide a complementary proof that when the number of voters is fixed, manipulation is in $\mathcal{P}$, for a broad class of other scoring elimination protocols (including `STV`).

Notably, we extend Hemaspaandra and Hemaspaandra's (2005) result from scoring protocols to scoring elimination protocols, providing a complete classification of such protocols into those that are easy and hard to manipulate for the weighted voters, bounded candidates case.

Finally we experimentally consider the proportion of `STV` instances for which it is clear that no manipulation is possible, to determine if complexity is really much of a good measure of difficulty for manipulation anyway. Although intuitively this may seem like a large proportion of instances, we find that it is not, providing some validation for the complexity approach.

2 Definitions

We adopt the convention that there are n candidates and m voters.

Definition 1. A **Vote** v on a set C (of size n) is an bijective function $v : C \to [n]$, that is, a function that assigns a value between 1 and n to each candidate.

We denote the set of all possible votes on C as $\mathcal{V}_C$.

Alternately we also sometimes consider a vote a *permutation* on or a *total ordering* of the set. We represent the ordering induced by a vote v in this fashion:

$$c_1 \succ c_2 \succ \ldots \succ c_n$$

implying that $v(c_1) = n, v(c_2) = n-1, \ldots, v(c_n) = 1$.

Note that this is the reverse of the usual way we think about votes (where the most preferred candidate gets first place).

Definition 2. Let $v \in \mathcal{V}_C$ be a vote on the candidates C and let $C' \subset C$ be a subset of those voters. Then $v|_{C'} \in \mathcal{V}_{C'}$ is v restricted to C' in the obvious way (by the induced total ordering on C'). We define the restriction of V to C', $V|_{C'} = \{v|_{C'}, v \in V\}$

Definition 3. An **Election** consists of a set C (known as the candidates) and a set V of votes on C. It is denoted (C, V).

Definition 4. A **Voting Protocol** `Prot` is a function, which, given an election (C, V) returns some $c \in C$, the winner of that election. Further, it is required that `Prot` is in $\mathcal{P}$, that is, we can evaluate `Prot` in time polynomial in n and m. We say $\texttt{Prot}(C, V) = c$.

Definition 5. Let $\tilde{V}$ be a set of non-negative integers of size $n = |V|$. We define the **Weighted Election** $(C, V, \tilde{V})$ to be the election amongst C with each vote $v_i \in V$ weighted by the corresponding weight $\tilde{v_i}$.

Given any voting protocol `Prot`, we say

$$\texttt{Prot}(C, V, \tilde{V}) = \texttt{Prot}(C, V')$$

Where $V' = \{\tilde{v_i}$ copies of $v_i, v_i \in V\}$.

Definition 6. A Voting Protocol `Prot` is **manipulable** if there exists an election $(C, V \uplus \{v\})$,[2] in which $\forall c \neq p, v(p) > v(c)$, but $\texttt{Prot}(C, V \uplus \{v\}) \neq p$, and there is some vote v' such that $\texttt{Prot}(C, V \uplus \{v'\}) = p$. The vote v' is referred to as a **manipulation** for the vote v. Candidate p is the *preferred candidate* in vote v.

Voting Protocols

In this section we outline exactly the voting protocols that we will refer to in this paper.

`Plurality`

Given an election (C, V), we declare the winner to be the candidate in C with the most first preference votes in V.

`Veto`

Given an election (C, V), we declare the winner to be the candidate in C with the least last preference votes in V.

Single Transferable Vote (`STV`)

`STV` is described in the following algorithm. We define the winner of the election (C, V) to be the winner of the election $(C', V|_{C'})$, where C' is the set of candidates remaining after eliminating c, the candidate who receives the least first preference votes.

Function `SingleTransferableVote`

input : An Election (C, V)
output: The elected candidate c

if $|C| = 1$ **then return** $c \in C$

Initialise the tallies
forall $c \in C$ **do** $\mathsf{Tally}[c] = 0$

Distribute the votes
forall $v \in V$ **do** $\mathsf{Tally}[\texttt{Head}(v)]$ += 1

Eliminate candidate with lowest tally.
$c = \arg\min_{r \in R} \mathsf{Tally}[r]$
$C' = C \setminus \{c\}$

Recursive Call
return `SingleTransferableVote` $(C', V|_{C'})$

Here `Head` is the function to take the first preference from a vote.

Notice that we do not specify what happens in these elimination protocols when the scores are tied. The problem is that there is no obvious "fair" system of tie-break. Many different systems are used to resolve ties. For the purposes of this paper we will always assume that ties go against the manipulator: that is, it is not good enough for a manipulator to ensure a tie to influence an elimination (or equivalent for some other protocol), she must make sure that the given candidate wins or loses by at least a single vote.

[2] Note the use of $\uplus$ to highlight the fact our "set" of votes isn't really a set, but in fact an ordered multiset. Thus when we take the union, we are necessarily taking a *disjoint union*.

Some Manipulation Problems

The manipulation problems we will be focusing on are the following— successive generalisations of each other.

SIMPLE MANIPULATION

GIVEN: A set C of candidates with a preferred candidate p, and a set V of known votes.
QUESTION:
Is there a vote w such that $\texttt{Prot}(C, V \uplus \{w\}) = p$?

COALITIONAL MANIPULATION

GIVEN: A set of candidates C, a set V of known votes, a number k of votes to be determined, and a preferred candidate $p \in C$
QUESTION: Is there a set of votes W of size k such that the $\texttt{Prot}(C, V \uplus W) = p$?

WEIGHTED COALITIONAL MANIPULATION(WCM)

GIVEN: A set C of candidates with a preferred candidate p, and a set V of known voters, as set of corresponding weights $\tilde{V}$. A set of weights $\tilde{W}$ of size k.
QUESTION: Is there a set of votes W (of size k), such that the weighted election $(C, V \uplus W, \tilde{V} \uplus \tilde{W})$ results in the election of p?

Note that our requirement in Definition 4 that evaluating `Prot` is in $\mathcal{P}$ guarantees that all problems are in $\mathcal{NP}$.

3 Scoring Protocols

Scoring Protocols (also known as a Positional Protocols) are a class of voting protocols that can be characterised by associating a score with each preference on a voting ballot. We decide an election by choosing which candidate has the highest total after tallying the scores from each of the votes.

The `Borda` protocol is the prototypical scoring protocol. For an election with n candidates, given any vote, we assign no points to the last preference, one to the second preference, up to $n-1$ points to the first preference. We can represent this protocol by the vector $(n-1, n-2, \ldots, 1, 0)$, where each number represents how many points it is worth to be in that position on any given vote.

We have seen some other examples of scoring protocols already. The `Plurality` protocol can be implemented as a scoring protocol with score vector $(1, 0, \ldots, 0)$, and the `Veto` protocol by $(1, 1, \ldots, 1, 0)$.

Definition 7. A **Score Vector** on n candidates, is some vector

$$\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)$$

with each $\alpha_i \in \mathbb{Z}$, subject to the conditions $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$ and not all α_i equal.

Definition 8. Given some score vector $\vec{\alpha}$ on n candidates, and some vote $v \in \mathcal{V}_\mathcal{C}$, we define the **Score** of some candidate $c \in C$ from v parametrised by $\vec{\alpha}$:

$$v_{\vec{\alpha}}(c) = \alpha_{n-v(c)+1}$$

We define the score of c under some set of votes V parametrised by $\vec{\alpha}$:

$$V_{\vec{\alpha}}(c) = \sum_{v \in V} v_{\vec{\alpha}}(c)$$

Definition 9. A Voting Protocol `Prot` is a **Scoring Protocol** if there exists, for each n, a score vector $\vec{\alpha}$ of size n such that for any election on n candidates, (C, V),

$$\texttt{Prot}(C, V) = \arg\max_{c \in C} V_{\vec{\alpha}}(c)$$

To be precise we require that there exists a polynomial time computable "scoring" function S from natural numbers n (represented in unary) to score vectors of size n. We denote the protocol defined by such a scoring function S by $\texttt{Prot}_S$.

Such a scoring protocol will rank candidates from first to last (which was not a requirement of our general voting techniques, but which will be useful later).

Definition 10. We say a score vector $\vec{\alpha}$ is in **Normal Form** if $\alpha_n = 0$ and $\gcd\{\alpha_i | \alpha_i \neq 0\} = 1$.

Likewise a Scoring function S or Scoring Protocol $\texttt{Prot}_S$ is in Normal Form if $S(n)$ is in normal form for all n.

So the score vectors given for the `Borda`, `Plurality` and `Veto` protocols given above were all in normal form.

The following results establishes the existence and uniqueness of a normal form for any protocol:

Proposition 1 (Hemaspaandra & Hemaspaandra (2005, Observation 2.2))**.** *Let $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)$ be a scoring vector. Then for all elections (C, V), (of size n) and candidates $c_1, c_1 \in C$ the following hold:*

- *For all integers k, $\vec{\alpha}+k = (\alpha_1+k, \alpha_2+k, \ldots, \alpha_n+k)$ is a scoring vector, and $\texttt{Prot}_{\vec{\alpha}}$ ranks c_1 above c_2 if and only if $\texttt{Prot}_{\vec{\alpha}+k}$ ranks c_1 above c_2.*
- *For all positive integers k, $k\vec{\alpha} = (k\alpha_1, k\alpha_2, \ldots, k\alpha_n)$ is a scoring vector, and $\texttt{Prot}_{\vec{\alpha}}$ ranks c_1 above c_2 if and only if $\texttt{Prot}_{k\vec{\alpha}}$ ranks c_1 above c_2.*

Manipulating Scoring Protocols

We begin by establishing some results about which scoring protocols are known to be easy or hard for which problems.

Definition 11. A scoring protocol $\texttt{Prot}_S$ is a **Binary Scoring Protocol** if for all n, $S(n) = (\alpha_1, \ldots, \alpha_n)$, all α_i coordinates are either 0 or 1.

For instance `Plurality` and `Veto` are Binary, but `Borda` is not.

Theorem 2. SIMPLE MANIPULATION *is in $\mathcal{P}$ for any Scoring Protocol $\texttt{Prot}_S$.*

Proof. Let $\vec{\alpha} = S(n)$. It is clear that candidate p should be placed in the highest position of our vote. So p will have final score $V_{\vec{\alpha}}(p) + \alpha_1$.

Rank all other candidates in terms of $V_{\vec{\alpha}}$ (score from the votes in V), from lowest to highest as $\{c_1, c_2, \ldots, c_{n-1}\}$. We now place the candidates in the remaining positions in that order (that is c_1 in position 2, etc).

Suppose that placing some c_i in position $i+1$ means it beats p. So the only way to make sure p wins is to place c_i in a higher (and thus lower scoring) position. But there are only $n-(i+1)$ such positions, and we know there are $n-i$ candidates with at least as high a score as c_i.

So we will have to place one of the $n-i$ candidates $\{c_i, \ldots, c_{n-1}\}$, each who have equal or higher score than c_i, in position $i+1$ or lower, say position $k \leq i+1$. But as $\alpha_k \geq \alpha_{i+1}$, this candidate will now beat p. So no solution was possible. □

Theorem 3. COALITIONAL MANIPULATION *is in* $\mathcal{P}$ *for any Binary Scoring Protocol* $\texttt{Prot}_S$.

Proof. Let $\vec{\alpha} = S(n)$. We begin in the same fashion, placing p in the top position of each vote. We will now be able to give each $c \in C$ at most $\texttt{slack}(c)$ points, where:

$$\texttt{slack}(c) = V_{\vec{\alpha}}(p) + k\alpha_1 - V_{\vec{\alpha}}(c) - 1$$

That means we will have to place c in at least $k - \texttt{slack}(c)$ 0 positions. If this number is is greater than k, we cannot stop c from beating p and we stop.

We place the c's one at a time, in any order. For each c we choose the $k - \texttt{slack}(c)$ votes with the most remaining 0's, and place c in any position worth 0 points. We then place c in a 1 position in each other vote. This guarantees that no vote will have more than one more 0 position than any other at any stage.

Suppose for some candidate we do not have enough votes with remaining 0 positions. Then as some votes have no 0 positions, the most a single vote can have is one. Thus there are not enough 0 positions left. So as we were always giving the candidates the least possible 0's, no solution exists. □

Theorem 4 (Hemaspaandra and Hemaspaandra (2005))**.** *For any scoring protocol* $\texttt{Prot}_S$*, for any* n*, if* $S(n) = (\alpha_1, \alpha_2, \ldots, \alpha_n)$*,* WCM *is in* $\mathcal{P}$ *if* $\alpha_2 = \alpha_3 = \cdots = \alpha_n$*, and is* $\mathcal{NP}$*-complete otherwise.*

So this result says that any scoring protocol bar `Plurality` is $\mathcal{NP}$-hard to manipulate with more than three candidates (any scoring protocol is equivalent to `Plurality` with two candidates anyway), and `Plurality` is in $\mathcal{P}$. In Section 4 we show the equivalent result for our generalisation of `STV`, Scoring Elimination Protocols.

4 Scoring Elimination Protocols

As part of outlining a technique for hybridising voting protocols, Elkind and Lipmaa (2005) discuss the concept of the **Closure** of a protocol. That is, given some protocol `Prot` which ranks the candidates (as we've seen scoring protocols do), we construct a new protocol $\overline{\texttt{Prot}}$.

Definition 12. Let `Prot` be any protocol which can be used to rank all candidates. Then $\overline{\texttt{Prot}}$, the **Closure** of `Prot`, is defined in the following recursive fashion:

Let $\overline{\texttt{Prot}}(\{c\}, V) = c$. Let $\overline{\texttt{Prot}}(C, V) = \overline{\texttt{Prot}}(C', V|_{C'})$, where $C' = C \setminus \{c\}$, and c is the candidate ranked last by `Prot` on the election (C, V).

The closure of a protocol has a very familiar structure—this is because in fact $\overline{\texttt{Plurality}} = \texttt{STV}$. We are interested in general properties of the closure of scoring protocols as generalisations of `STV`. We call the closure of a scoring protocol a **Scoring Elimination Protocol**.

Manipulating Scoring Elimination Protocols

It has been shown (Bartholdi & Orlin 1991) that `STV` is hard to manipulate in the general case. As `STV` is one of the simplest scoring elimination protocols, this indicates most are likely to be difficult to manipulate in the general case. We consider cases where the problem is limited in some way.

Fixed Candidates As with any other voting protocol, COALITIONAL MANIPULATION is easy for scoring elimination protocols when the number of candidates is constant (Conitzer & Sandholm 2002).

However the algorithm described requires checking up to $\binom{n!+k-1}{k}$ different elections (where k is the number of voters in a manipulating coalition), which is very large, even for quite small n and k.

To get an idea, for example with $n = 5, k = 10$, we have 2.5×10^{14} possible votes profiles. The situation is much worse when the number of candidates grows—when $n = 8, k = 10$, we have a staggering 3.1×10^{39} choices of vote profile, which is more than the total number of keys in a 128 bit encryption algorithm! Clearly this is not a feasible algorithm in such cases.

We outline a more efficient algorithm below:

Algorithm 2: COALITIONAL MANIPULATION

```
data : votes: k empty, unmarked votes
forall Elimination orders (c_1, ..., c_n = p) do
2.1   forall c_i in {c_1, ..., c_{n-1}} do
          Tally the current votes for {c_i, ..., c_n}
          forall c_j in {c_{i+1}, ..., c_n} do
              while c_j has less votes than c_i do
                  if ∃ v ∈ votes unmarked then
2.2                   Mark v, place c_j in highest
                      remaining position
                  else
2.3                   Elimination order is
                      impossible
                      Try next elimination order
          Now "eliminate" c_i
          forall marked votes v in votes do
              if v's lowest cand. is c_i then
                  Unmark v
      We have found a successful manipulation
      Fill all votes in votes in order: (p, c_{n-1}, ...)
      Return YES
  We have found no successful manipulation
  Return NO
```

The algorithm has a single data structure: the vote profile we are constructing (`votes`), where each vote at any point is marked "used" or "unused", and consists of an incomplete order (in the sense that it will have only the top $x < k$ positions filled). We begin with k completely unspecified, unmarked votes in `votes`.

The algorithm will investigate $(n-1)!$ potential elimination orders, the checking of each which is $\mathcal{O}(n(m + nk))$ in time. So the algorithm has $\mathcal{O}(n!(m + nk))$ time complexity (linear in m and k and only factorial in n, which is reasonable for low values).

It is clear that if the algorithm finds a manipulation, that manipulation will work – the algorithm clearly steps out the elimination procedure that will result from the election $(C, V \uplus W)$.

It is not so clear that if the algorithm fails to find a manipulation, none is possible. The following definitions will allow us to establish that result.

Definition 13. Given some fixed elimination order $(c_1, \ldots, c_n)$ of the candidates, the ownership map $\psi_i : \mathcal{V}_{\mathcal{C}} \to C$, defines to which candidate a vote "belongs" to at stage i of the elimination. (That is, just prior to the ith candidate being eliminated). It is defined:

$$\psi_i(v) = \arg\max_{\substack{c_j \in C \\ j \geq i}} v(c_j)$$

Definition 14. Given some manipulation W which induces an elimination order $(c_1, \ldots, c_n)$ of the candi-

dates; at a stage of the elimination i, the "tally" of a candidate c, denoted $\mathbf{W}_i(c)$ is defined in the obvious way:

$$\mathbf{W}_i(c) = |\{v \in W \uplus V, \psi_i(v) = c\}|$$

Note that the ownership map is necessarily undefined on votes marked unused at point i. We can think as the unused vote as not contributing to the tally of any candidate at that point. Also note that the tally is an increasing function for any c_j with i, whilst $i \leq j$.

Lemma 5. *For any fully specified manipulation W (that is, without unused votes), at any stage i in the elimination induced by W,*

$$\sum_{c \in C} \mathbf{W}_i(c) = |W| + |V|$$

Proof.

$$\begin{aligned}\sum_{c\in C} \mathbf{W}_i(c) &= \sum_{c\in C} |\{v \in W \uplus V, \psi_i(v) = c\}| \\ &= \sum_{c\in C} \sum_{\substack{v\in W\uplus V \\ \psi_i(v)=c}} 1 \\ &= \sum_{v\in W\uplus V} \sum_{\substack{c\in C \\ \psi_i(v)=c}} 1 = \sum_{v\in W\uplus V} |\mathrm{Image}(\psi_i(v))| \\ &\text{as } \psi_i \text{ is a well defined function on } W \uplus V \\ &= \sum_{v\in W\uplus V} 1 = |W \uplus V| = |W| + |V| \end{aligned}$$

□

The next Lemma shows that Algorithm 2 will in a sense minimize the number of votes each candidate has at any stage of the elimination.

Lemma 6. *Let $e = (c_1, \ldots, c_n = p)$ be a elimination order, and W a manipulation such that $(C, W \uplus V)$ induces e. Let i be some stage of that manipulation, and let X be the manipulation produced after the ith elimination loop (Point 2.1) of Algorithm 2 when attempting to guarantee e. Then for any candidate c, we have*

$$\mathbf{X}_i(c) \leq \mathbf{W}_i(c)$$

Proof. By induction on the candidates of e.

If c has been eliminated by stage i, then clearly $\mathbf{X}_i(c) = \mathbf{W}_i(c) = 0$. Suppose on the contrary that c has not been eliminated.

Let `prev` be the point at which the algorithm last assigned a vote to c (at Point 2.2).

If there is no such point it is clear that $\mathbf{X}_i(c)$ is equal to the the number of votes c would have received from V alone at stage i. As W cannot take votes away from a candidate, the inequality holds.

Suppose there is such a point; then the candidate that was being eliminated at stage `prev`, that is $c_{\texttt{prev}}$, must be lower in e than c (as we assume c has not yet been eliminated by stage i).

So we can assume by induction that

$$\mathbf{X}_{\texttt{prev}}(c_{\texttt{prev}}) \leq \mathbf{W}_{\texttt{prev}}(c_{\texttt{prev}})$$

But as W induces e, $c_{\texttt{prev}}$ is the candidate eliminated at point `prev`, so

$$\mathbf{W}_{\texttt{prev}}(c) > \mathbf{W}_{\texttt{prev}}(c_{\texttt{prev}}) \geq \mathbf{X}_{\texttt{prev}}(c_{\texttt{prev}})$$

But as the algorithm needs to give c a vote stage `prev`, we must have that

$$\mathbf{X}_{\texttt{prev}}(c) = \mathbf{X}_{\texttt{prev}}(c_{\texttt{prev}}) + 1$$

So we can see that, by combining the last two equations,

$$\mathbf{W}_{\texttt{prev}}(c) \geq \mathbf{X}_{\texttt{prev}}(c)$$

But `prev` was the last point the algorithm contributed to c's tally, so

$$\mathbf{X}_i(c) = \mathbf{X}_{\texttt{prev}}(c)$$

Also $\mathbf{W}_i(c)$ is increasing with i, and $i \geq \texttt{prev}$, so indeed

$$\mathbf{W}_i(c) \geq \mathbf{W}_{\texttt{prev}}(c) \geq \mathbf{X}_{\texttt{prev}}(c) = \mathbf{X}_i(c)$$ □

We are now able to prove that if a solution to Coalitional Manipulation exists for a given instance, Algorithm 2 will not fail.

Proof of Validity of Algorithm 2. Suppose that there is a manipulation W possible, but the algorithm fails to find a solution.

Clearly the election $(C, V \uplus W)$ will lead to some elimination order $(c_1, \ldots, c_n)$. Clearly as W leads to p winning, $c_n = p$, and so the algorithm will have investigated this order. So the algorithm must have broken at Point 2.3. Suppose we were ready to eliminate candidate c_i and we were unable to ensure the safety of candidate c_j.

Denote the set of votes constructed at stage i of the algorithm X. Note that as we have no remaining unused votes, ψ_i is well defined on X.

Clearly W ensures the safety of c_j at this point—otherwise it would induce a different elimination order.

So we have:

$$\mathbf{W}_i(c_i) < \mathbf{W}_i(c_j)$$

But we failed to ensure c_j's safety, so

$$\mathbf{X}_i(c_j) \leq \mathbf{X}_i(c_i)$$

Now by Lemma 6,

$$\mathbf{X}_i(c_i) \leq \mathbf{W}_i(c_i)$$

So we conclude that

$$\mathbf{X}_i(c_j) < \mathbf{W}_i(c_j) \tag{1}$$

Given that $|W| = |X| = k$, by Lemma 5, (1) implies that

$$\mathbf{W}_i(c_k) < \mathbf{X}_i(c_k)$$

for some c_k. But this is impossible by Lemma 6. □

Note that this algorithm could be improved by the use of backtracking: that is, if we are investigating the elimination order $(c_1, c_2, \ldots, c_n = p)$ and we fail ensure that c_i is eliminated (Point 2.3), then there is no point in investigating any other elimination order of the form $(c_1, c_2, \ldots, c_i, \ldots)$.

Fixed Voters We consider the opposite problem: when the number of voters is fixed (and the number of candidates is allowed to grow unbounded). We see that a similar easiness result holds for a broad class of Scoring Elimination protocols (including STV):

Definition 15. A Scoring (Elimination) Protocol $\texttt{Prot}_S$ is a **Finite Selection** Scoring (Elimination) Protocol if there exists a constant K such that for all $n \geq K$, if $S(n) = (\alpha_1, \ldots, \alpha_n)$, then $\alpha_K > 0$ but $\alpha_{K+1} = \cdots = \alpha_n = 0$.

Theorem 7. *Let* Prot *be a Finite Selection Scoring Elimination Protocol. Then, for a constant number of voters,* Coalitional Manipulation *is in* $\mathcal{P}$ *for* Prot.

To prove this theorem we use the following simple fact:

Lemma 8. *Suppose that* Prot *is a Finite Selection Scoring Elimination protocol, with a given K-value. Then given any election (C, V), there can be at most Km candidates C' whom are ranked at a Kth or higher position in some vote in V and* $\texttt{Prot}(C, V) = \texttt{Prot}(C', V|_{C'})$.

Proof. The first statement is clear by the definition of Finite Selection Protocols. The second follows by induction after considering the effect of elimination of a candidate not in C'. Clearly one of these candidates will be eliminated first as by definition they will have a score of zero, and any candidate in C' will have at least a single point.

As any $c \notin C'$ is not in the top K positions of any vote, elimination of c will not promote any other candidate into such a position. Thus elimination of c will not affect any candidates' score, or the composition of C'. So by induction, all candidates not in C' will be eliminated first. □

Proof of Theorem 7. To begin with, we can assume that $k \leq m$. Otherwise we can simply reverse[3] all votes in V and make all others $p \succ \cdots$ (everything else arbitrary). Thus k is bounded.

Now suppose we are given the instance (C, V, p). Let C' be the set of candidates voted at or above Kth position by V. We now must ourselves choose at most Kk candidates to place at or above the Kth position. There are less than n^{Kk} (both k and K are bounded) such choices.

Now call C'' the set of candidates voted at or above the Kth position by both V and our manipulation W. By Lemma 8, the election will now be determined by $W|_{C''}$—that is the ordering of the remaining candidates in $C \setminus C''$ by our votes is irrelevant.

So we only have to worry about ordering at most $(k + m)K$ candidates. So we only have less than $((k + m)K)!n^{Kk}$ choices that will affect the election differently. We can thus simply check each. □

Hardness Results for Scoring Elimination Protocols

We know (Theorem 4) that even with fixed candidates, WCM is hard for almost all Scoring Protocols. We also know (Conitzer & Sandholm 2002) that it is hard for STV with fixed candidates. We will use a reduction from WCM on Scoring Protocols to show that it is indeed hard for almost all Scoring Elimination Protocols.

To do this we consider that manipulating a protocol $\overline{\texttt{Prot}}$ involves influencing who is *eliminated* in some stage of the election. This involves making a candidate come last in a single instance of Prot. With this in mind, we define:

Anti-WCM

GIVEN: A set C of Candidates with a disliked candidate d, and a set V of known voters, as set of corresponding weights $\tilde{V}$. A set of weights $\tilde{W}$ of size k.
QUESTION: Is there a set of votes W (of size k), such in that the weighted election $(C, V \uplus W, \tilde{V} \uplus \tilde{W})$, d will received the lowest score?

The Adjoint of a Scoring Protocol

We will need to know exactly which Scoring Protocols Anti-WCM is hard for. In order to establish this, we will need the concept of the adjoint.

One thing we notice about the Plurality and Veto protocols is that in a sense they are opposite—Suppose candidate p gets the highest score under Plurality voting over a vote profile V; then on the profile with all votes reversed (e.g. all first preferences last), p will get the lowest score under Veto.

This concept is defined more clearly in the following:

Definition 16. Given a vote v, the **adjoint vote** v^* is the vote which induces the opposite ordering on the candidates. So if v induces the ordering $c_n \succ \cdots \succ c_1$, then v^* induces $c_1 \succ \cdots \succ c_n$

Mathematically this means

$$v^*(c) = n + 1 - v(c)$$

We also write $V^* = \{v^*, v \in V\}$.

Definition 17. Given a scoring vector $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)$ the **Adjoint Vector** is defined:

$$\vec{\alpha}^* = (\alpha_1 - \alpha_n, \ldots, \alpha_1 - \alpha_1)$$

Given a scoring protocol $\texttt{Prot}_S$, we define the **Adjoint Protocol**, $\texttt{Prot}^*_S = \texttt{Prot}_{S^*}$, where $S^*(n) = S(n)^*$.

Note that given a scoring vector in normal form, the construction will guarantee the adjoint is also in normal form. Note also that taking the adjoint is an involution. That is $v^{**} = v$ and $\vec{\alpha}^{**} = \vec{\alpha}$. For example $\texttt{Plurality}^* = \texttt{Veto}$, and that $\texttt{Borda}^* = \texttt{Borda}$.

The adjoint is in a sense the opposite: on the opposite set of votes (V^*), it will rank candidates in the opposite order.

Proposition 9. *Let* Prot *be a scoring protocol, and* Prot* *its adjoint. Let $c_1, c_2 \in C$, for some election (C, V). Then* Prot *will rank c_1 higher than c_2 on the election (C, V) if and only if* Prot* *ranks c_1 lower than c_2 on (C, V^*).*

Proof. Consider the score of c_i under the adjoint of a vote in the adjoint of a protocol:

$$\begin{aligned} v^*_{\vec{\alpha}^*}(c_i) &= \alpha^*_{n-v^*(c_i)+1} = \alpha_1 - \alpha_{n-(n-v^*(c_i)+1)+1} \\ &= \alpha_1 - \alpha_{v^*(c_i)} = \alpha_1 - \alpha_{n-v(c)+1} \\ &= \alpha_1 - v_{\vec{\alpha}}(c) \end{aligned}$$

So we have that

$$V^*_{\vec{\alpha}^*}(c_i) = m\alpha_1 - V_{\vec{\alpha}}(c_i)$$ □

Now the we have the requisite concepts, we can show:

[3] Note that for any elimination protocol, a pair of reverse votes simply cancel each other out, regardless of elimination order.

Lemma 10. WCM *is $\mathcal{NP}$-hard for a scoring protocol* `Prot` *if and only if* ANTI-WCM *is $\mathcal{NP}$-hard for the adjoint protocol* $\texttt{Prot}^*$.

Proof. This follows very easily from Proposition 9. To reduce from an instance $(C, p(d), V, \tilde{V}, \tilde{W})$ of WCM (ANTI-WCM), we simply output the instance $(C, p(d), V^*, \tilde{V}, \tilde{W})$ of ANTI-WCM (WCM). □

Corollary 10.1. *For any scoring protocol* $\texttt{Prot}_{\vec{\alpha}}$, $\vec{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$, ANTI-WCM *is in P if $\alpha_1 = \alpha_2 = \cdots = \alpha_{n-1}$, and is $\mathcal{NP}$-complete otherwise.*

Main Result for Scoring Elimination Protocols

Corollary 10.1 has shown us that for any (non-`Veto`) scoring protocol, it is difficult to influence which candidate will be ranked last. It would seem natural to try and extend this result to scoring elimination protocols, manipulation of which involves deciding which candidates will be ranked last.

We begin by showing that the scoring protocol that is the closure of `Veto`, the `Coombs` protocol (Straffin Jr. 1980), is easy to manipulate when the number of candidates is fixed (even when the controlled votes are weighted!)

Theorem 11. *On the* $\texttt{Coombs}\ (= \overline{\texttt{Veto}})$ *protocol,* WCM *is in $\mathcal{P}$, assuming the number of candidates is fixed.*

To prove this theorem, we need the following Lemma:

Lemma 12. *Let E be an instance of* WCM *on the* `Coombs` *protocol, and e elimination order, $e = (c_1, c_2, \ldots, c_n)$. Let w be the vote inducing*

$$c_n \succ c_{n-1} \succ \cdots \succ c_1$$

Then if there exists any set of votes W inducing e, the set of votes $X = \{k$ copies of $w\}$ induces e.

Proof. By induction. For the general case, it is clear that X gives at least as much last place weight to c_1 as W. So as W results in c_1's elimination, X clearly does. After c_1 is eliminated we will consider the election on $n-1$ candidates, with all votes in X of the form:

$$c_n \succ c_{n-1} \succ \cdots \succ c_2$$

But we know by induction that this set will guarantee e. □

Proof of Theorem 11. Armored with Lemma 12, we simply try each elimination order $(c_1, \ldots, c_n = p)$. The lemma tells us that we only need to try one W (the X above) to ascertain if the given elimination order is possible. There are $(n-1)!$ such elimination orders, as n is constant, we can simply test each one. □

Now we know `Coombs` is in $\mathcal{P}$ for WCM, we investigate protocols different to `Coombs`.

Theorem 13. *Let* $\overline{\texttt{Prot}_S}$ *be the closure of a scoring protocol. Suppose $S(n) = (\alpha_1, \alpha_2, \ldots, \alpha_n)$. Then* WCM *is (when the number of candidates is bounded) in $\mathcal{P}$ if $\alpha_1 = \alpha_2 = \cdots = \alpha_{n-1}$ for all n, and is $\mathcal{NP}$-hard otherwise.*

Proof. Note that if the condition holds, $\overline{\texttt{Prot}_S} = \texttt{Coombs}$ which by Theorem 11, is in $\mathcal{P}$.

Suppose that the condidition does not hold. Let n be the smallest value for which it does not. (So $n \geq 3$).

We will construct a reduction from ANTI-WCM on $\texttt{Prot}_S$ on n candidates. Given an instance with candidates $c_1, \ldots, c_n$, assume without loss of generality that the disliked candidate is c_n. Add the following cyclical permutation of votes X to V, each with equal weight K in $\tilde{V}$ greater than the sum of the original weights in $\tilde{V}$ and $\tilde{W}$ combined.

$$\begin{aligned} c_1 \succ c_2 \succ \cdots \succ c_n \\ c_2 \succ \cdots \succ c_n \succ c_1 \\ c_3 \succ \cdots \succ c_n \succ c_1 \succ c_2 \\ \vdots \\ c_n \succ c_1 \succ \cdots \succ c_{n-1} \end{aligned}$$

Set c_1 as the preferred candidate.

Note that all candidates will receive equal points from X, whatever the protocol. Consider which candidate is eliminated next. Suppose it is c_i for some $1 \leq i \leq n$.

After the elimination of c_i the votes in X will constitute another cyclical permutation on the remaining $n-1$ candidates, however there will now be two votes of the form:

$$c_{i+1} \succ c_{i+2} \succ \cdots \succ c_{i-1}$$

(Here $c_{n+1} = c_1$)

Now by assumption, as n was the smallest value at which our protocol was not equivalent to `Coombs`, from now on our protocol will eliminate the candidate with the most last preferences. X has given c_{i-1} K extra last preferences, which is more than the total sum of last preferences in V and W. So c_{i-1} will certainly be eliminated next.

After c_{i-1} is eliminated we will again have in X a cyclical permutation of the candidates, of which there are now $n-2$. But now we will have an 3 votes of the form:

$$c_{i+1} \succ c_{i+2} \succ \cdots \succ c_{i-2}$$

Continuing this argument by induction we see that next c_{i-2} is eliminated, then c_{i-3}, until c_{i+1} is the eventual winner. So c_1 will win the election if and only if c_n is the first candidate eliminated. That is there exists a solution to the instance of WCM on the open protocol if and only if a there exists a solution to the instance of WCM on the closed protocol. □

Clearly this result extends to unlimited candidates for the non-`Coombs` protocols. We leave the difficult of WCM on `Coombs` for unlimited candidates as an open question.

5 Empirical Results

It is common (and not unfair) to criticise the use of computational complexity as a measure of the difficulty of a problem. After all $\mathcal{NP}$-completeness simply tells us that there exist difficult instances of the problem, not that all or even many of the instances are difficult. Certainly the instances of `STV` that $\mathcal{NP}$-hardness proofs generate are hardly average.

With this in mind, we considered what proportion of the time is COALITIONAL MANIPULATION trivially impossible to manipulate for `STV`. We consider the question of trivially unmanipulable instances, defined as follows:

Definition 18. Let $E = (C, V, p, k)$ be an instance of COALITIONAL MANIPULATION. Suppose the application of `STV` to E induces the elimination order

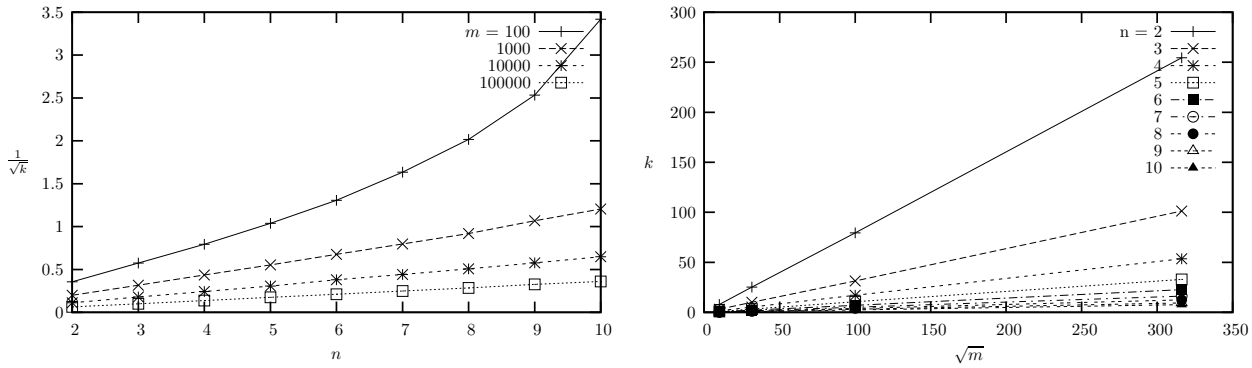

Figure 1: Graphs of mean minimum k required against number of candidates (n) and number of voters (m)

$e = (c_1, \ldots c_n)$. Then E is **Trivially Unmanipulable** if for all manipulations W (of size k), STV applied to $(C, V \uplus W)$ induces e.

This means that if the losing candidate in any round of the unmodified election's elimination always loses by more than k, there's no way that W can influence the order of elimination, let alone the winner of the election. Certainly such instances are polynomially recognizable.

In order to try and estimate the proportion of such trivially unmanipulable elections, we ran a series of experiments, generating elections of various sizes, and counting the minimum margin by which an eliminated candidate lost by (a manipulating coalition would need one more than this number to make the election not trivially unmanipulable).

For each n / m combination, we ran 10000 experiments consisting of m voters each with a random vote selected with uniform probability from the $n!$ possible votes. The following table lists the mean minimum margin (k) of elimination, as described above:

$n \backslash m$	**100**	**1000**	**10000**	**100000**
2	7.8779	25.2486	79.4196	254.4100
3	3.0341	9.9872	31.4037	101.2660
4	1.5856	5.2881	17.0930	53.4922
5	0.9269	3.2786	10.6053	32.9697
6	0.5859	2.1891	6.9488	22.6186
7	0.3744	1.5722	5.1262	16.2440
8	0.2459	1.1827	3.8909	12.4222
9	0.1557	0.8753	3.0156	9.4553
10	0.0856	0.6896	2.3846	7.7521

In Figure 1 we plot first $\frac{1}{\sqrt{k}}$ against n to see that (especially when the number of voters is high), this forms a close to linear relationship, indicating that this value k is likely proportional to $\frac{1}{n^2}$. In the second we plot k against $\sqrt{m}$ and find a very linear relationship, strongly suggesting that this value of k is proportional to $\sqrt{m}$. Both of these facts indicate that as n and m rise, the fraction of the total number of voters (m) required to avoid trivially unmanipulable election falls.

This data demonstrates evidence that in general, for random instances of STV, only relatively small coalitional blocks are needed to affect election eliminations. Of course although this does not give an accurate estimate on the number needed to *manipulate* an election, it does give a lower bound, and so could conceivably be used as a heuristic method to rule out many electoral instances as unmanipulable.

So we see that it appears that most electoral instances are not trivially manipulable (for a relatively small coalitional group) and so this heuristic would be very unsuccessful, especially for large elections.

6 Conclusions

Voting protocols are an important method of decision making for agent-based systems. Although there are many methods of collating votes, one very important differential is the difficulty of manipulating such methods. We know that such manipulation is always possible, however it seems reasonable in this context to measure difficulty in terms of computational complexity. With such a framework, have found some protocols easier to manipulate than others.

Results

In this paper we have seen that STV is hard to manipulate in general for the Coalitional Manipulation problem. However, we have provided an algorithm that will find a manipulation in a reasonable amount of time, even with a relatively large number of candidates. Also we have demonstrated that when the number of voters is small, manipulation is easy for STV and a class of protocols containing it.

Considering generalisations of STV to more complicated scoring techniques, we have established the important result that Weighted Coalitional Manipulation is hard for any scoring elimination protocol—bar the Coombs protocol—for any number of candidates high enough to differentiate that protocol from Coombs.

Future Work

Although we have seen that all reasonable voting protocols are easy to manipulate for the simple question of Coalitional Manipulation with limited number of candidates, it is an open question as to the difficulty of manipulating many general protocols (and in particular scoring protocols) when the number of candidates is not limited.

It would be of particular interest to know more of the manipulation properties of scoring elimination protocols, as they can be more interesting than the scoring protocols they came from. For instance $\overline{\texttt{Borda}}$ is known as Nanson's protocol (Straffin Jr. 1980) and is Condorcet Consistent, whilst Borda is not. Although we have established that they are hard to manipulate in the Weighted Coalitional Manipulation case, it remains an open question as to whether they are hard to manipulate in the simpler case of Coalitional Manipulation (with unlimited candidates).

References

Bartholdi, J. J. & Orlin, J. A. (1991), 'Single transferable vote resists strategic voting', *Social*

Choice and Welfare **8**(4), 341–354.
URL: *http://www2.isye.gatech.edu/ jjb/papers/stv.pdf*

Colebatch, T. (2004), 'How party preferences picked family first', The Age.

Conitzer, V. & Sandholm, T. (2002), 'Complexity of manipulating elections with few candidates'.
URL: *citeseer.ist.psu.edu/conitzer02complexity.html*

Elkind, E. & Lipmaa, H. (2005), Hybrid Voting Protocols and Hardness of Manipulation, *in* 'First Spain Italy Netherlands Meeting on Game Theory', Maastricht, The Netherlands. Workshop without proceedings. Webpage at http://www.fdewb.unimaas.nl/sing/.

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

Hemaspaandra, E. & Hemaspaandra, L. (2005), 'Dichotomy for voting systems'.
URL: *http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cs/0504075*

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

Straffin Jr., P. D. (1980), *Topics in the Theory of Voting*, Birkhäuser Boston, Inc., 380 Green Street, Cambridge, MA.