Multi-Winner Elections: Complexity of Manipulation, Control, and Winner-Determination

Ariel D. Procaccia and **Jeffrey S. Rosenschein** and **Aviv Zohar**
School of Engineering and Computer Science
The Hebrew University of Jerusalem
{arielpro,jeff,avivz}@cs.huji.ac.il

Abstract

Although recent years have seen a surge of interest in the computational aspects of social choice, no attention has previously been devoted to elections with multiple winners, e.g., elections of an assembly or committee. In this paper, we fully characterize the worst-case complexity of manipulation and control in the context of four prominent multi-winner voting systems. Additionally, we show that several tailor-made multi-winner voting schemes are impractical, as it is $\mathcal{NP}$-hard to select the winners in these schemes.

1 Introduction

Computational aspects of voting have been the focus of much interest, in a variety of fields. In multiagent systems, the attention has been motivated by applications of well-studied voting systems[1] as a method of preference aggregation. For instance, Ghosh et al. [1999] designed an automated movie recommendation system, in which the conflicting preferences a user may have about movies were represented as agents, and movies to be suggested were selected according to a voting scheme (in this example there are multiple winners, as several movies are recommended to the user). In general, the candidates in a virtual election can be entities such as beliefs or joint plans [Ephrati and Rosenschein, 1997].

Different aspects of voting rules have been explored by computer scientists. An issue that has been particularly well-studied is manipulation. The celebrated Gibbard-Satterthwaite Theorem implies that under any reasonable voting scheme, there always exist elections in which a voter can improve its utility by lying about its true preferences. Nevertheless, it has been suggested that bounded-rational agents may find it hard to determine exactly which lie to use, and thus may give up on manipulations altogether. The first to address this point were Bartholdi, Tovey and Trick [1989a]; Bartholdi and Orlin [1991] later showed that manipulating *Single Transferable Vote* (STV) is an $\mathcal{NP}$-complete problem. Conitzer, Lang and Sandholm [2003] studied a setting where there is an entire coalition of manipulators. In this setting, the problem of manipulation by the coalition is $\mathcal{NP}$-complete in a variety of protocols, even when the number of candidates is constant.

Another related issue that has received some attention is the computational difficulty of controlling an election. Here, the authority that conducts the elections attempts to achieve strategic results by adding or removing registered voters or candidates. Bartholdi, Tovey and Trick [1992] analyzed the computational complexity of these (and other) methods of controlling an election in the *Plurality* and *Condorcet* protocols.

The above discussion implies that computational complexity should be considered when contemplating voting systems that are seemingly susceptible to manipulation or control. On the other hand, taking into account computational costs can also lead to negative results. Some sophisticated voting systems, designed to satisfy theoretical desiderata, may be too difficult to use in real-world settings. In other words, there are voting systems where even determining who won the election is an $\mathcal{NP}$-complete problem. Previously known examples include voting schemes designed by Charles Dodgson[2] and Kemeny [Bartholdi *et al.*, 1989b].

Settings where there are multiple winners are inherently different from their single-winner counterparts. A major concern when electing an assembly, for example, might be *proportional representation*: the proportional support enjoyed by different factions should be accurately represented in the structure of the assembly. In practice, this usually means that the percentage of votes secured by a party is roughly proportional to the number of seats it is awarded.

Some simple multi-winner rules do not guarantee proportional results; these rules include *Single Non-Transferable Vote (SNTV)*, *Bloc voting*, *Approval*, and *Cumulative voting*. More recently, intriguing theoretical voting schemes have been devised with the goal of guaranteeing proportional representation. Two such schemes that have received attention were proposed, respectively, by Monroe [1995], and by Chamberlin and Courant [1983].

In this paper, we augment the classical problems of manipulation and control by introducing multiple winners, and study these problems with respect to four simple but impor-

[1] We use the terms "voting schemes", "voting rules", "voting systems", and "voting protocols" interchangeably.

[2] Better known as Lewis Carroll, author of "Alice's Adventures in Wonderland".

tant multi-winner voting schemes: SNTV, Bloc voting, Approval, and Cumulative voting; we find that Cumulative voting is computationally resistant to both manipulation and control. In addition, we characterize the computational complexity of winner determination in some of the intriguing voting schemes that have been suggested in recent years by political scientists.

2 Multi-Winner Voting Schemes

In this section we discuss several multi-winner voting systems of significance. Although the paper is self-contained, interested readers can find more details in [Brams and Fishburn, 2002].

Let the set of voters be $V = \{v_1, v_2, \dots v_n\}$; let the set of candidates be $C = \{c_1, c_2, \dots c_m\}$. Furthermore, assume that $k \in \mathbb{N}$ candidates are to be elected.

We first present four simple voting schemes; in all four, the candidates are given points by the voters, and the k candidates with the most points win the election. The schemes differ in the way points are awarded to candidates.

- *Single Non-Transferable Vote* (SNTV): each voter gives one point to a favorite candidate.
- *Bloc voting*: each voter gives one point to each of k candidates.
- *Approval voting*: each voter can approve or disapprove any candidate; an approved candidate is awarded one point, and there is no limit to the number of candidates a voter can approve.
- *Cumulative voting*: allows voters to express intensities of preferences, by asking them to distribute a fixed number of points among the candidates. Cumulative voting is especially interesting, since it encourages minority representation and maximizes social welfare [Brams and Fishburn, 2002].

2.1 Fully Proportional Representation

We now describe two theoretical voting schemes that attempt to achieve the ideal of fully proportional representation.

We begin by specifying Monroe's pure scheme [Monroe, 1995]. For each voter v and candidate c, a *misrepresentation value* μ_{vc} is known;[3] this value characterizes the degree to which candidate c misrepresents voter v.

Let $\mathcal{S} = \{S \subseteq C : |S| = k\}$, the set of all possible subsets of k winners. Let $S \in \mathcal{S}$, and let $f_S : V \to S$ be a function that assigns voters to candidates in S. The misrepresentation score of voter v under f_S is $\mu_{v f_S(v)}$. The total misrepresentation of assignment f_S is $\sum_{v \in V} \mu_{v f_S(v)}$.

Monroe requires that f_S be restricted so that a similar number of voters be assigned to each candidate in S. In other words, each candidate in S must be assigned at least $\lfloor n/k \rfloor$ voters. We say that such an assignment is *balanced*. The misrepresentation score of S is the misrepresentation score of f_S, where $f_S : V \to S$ is the assignment with the minimal misrepresentation, subject to the above restriction. The k winners are the set $S \in \mathcal{S}$ with the lowest misrepresentation score.

Chamberlin and Courant [1983] adopt a similar approach; as before, one considers sets $S \in \mathcal{S}$ and assignments f_S. However, Chamberlin and Courant impose no restrictions on the assignments. Therefore, each set S is associated with the assignment $f_S : V \to S$ that minimizes misrepresentation among all possible assignments. To maintain proportionality, Chamberlin and Courant compensate by using weighted voting in the assembly.

3 Manipulation

A voter is considered to be a *manipulator*, or is said to *vote strategically*, if the voter reveals false preferences in an attempt to improve its outcome in the election. Settings where manipulation is possible are to be avoided, as many voting protocols are designed to maximize social welfare, under the assumption that voters reveal their intentions truthfully. Therefore, computational resistance to manipulation is considered an advantage.

In the classical formalization of the manipulation problem [Bartholdi *et al.*, 1989a], we are given a set C of candidates, a set V of voters, and a distinguished candidate $p \in C$. We also have full knowledge of the voters' votes. We are asked whether it is possible to cast an additional vote, the manipulator's ballot, in a way that makes p win the election.

When generalizing this problem for the k-winner case, several formulations are possible. For example, one can ask whether some candidate can be one of the k-winners, or whether it is possible to ensure that a complete set of k winners be elected. We adopt a more general formulation.

Definition 1. In the MANIPULATION problem, we are given a set C of candidates, a set V of voters that have already cast their vote, the number of winners $k \in \mathbb{N}$, a utility function $u : C \to \mathbb{Z}$, and an integer $t \in \mathbb{N}$. We are asked whether the manipulator can cast its vote such that in the resulting election: $\sum_{c \in W} u(c) \geq t$, where W is the set of winners, $|W| = k$.

Remark 1. We make the standard assumption that tie-breaking is adversarial to the manipulator [Conitzer and Sandholm, 2002; Conitzer *et al.*, 2003], i.e., if there are several candidates that perform equally well in the election, the ones with the lower utility will be elected.

Proposition 1. MANIPULATION *in SNTV, Bloc voting, and Approval is in $\mathcal{P}$.*

Proof. Simple and efficient algorithms exist for MANIPULATION in these three protocols; omitted due to lack of space. □

Proposition 2. MANIPULATION *in Cumulative voting is $\mathcal{NP}$-complete.*

The proof relies on a reduction from one of the most well-known $\mathcal{NP}$-complete problems, the KNAPSACK problem.

Definition 2. In the KNAPSACK problem, we are given a set of items $A = \{a_1, \dots, a_n\}$, for each $a \in A$ a weight

[3] The misrepresentation values μ_{vc} may be naturally derived from ballots cast by the electorate, but we do not go into details as to exactly how this can be done. In any case, it is logical to assume that $\mu_{vc} \in \{0, 1, \dots, m\}$, and we make this assumption throughout the paper.

$w(a) \in \mathbb{N}$ and a value $v(a)$, a capacity $b \in \mathbb{N}$, and $t \in \mathbb{N}$. We are asked whether there is a subset $A' \subseteq A$ such that $\sum_{a \in A'} v(a) \geq t$ while $\sum_{a \in A'} w(a) \leq b$.

Proof of Proposition 2. The problem is clearly in $\mathcal{NP}$.

To see that MANIPULATION in cumulative voting is $\mathcal{NP}$-hard, we prove that KNAPSACK reduces to this problem. We are given an input $\langle A, w, v, b, t \rangle$ of KNAPSACK, and construct an instance of MANIPULATION in Cumulative voting as follows.

Let n=$|A|$. There are $2n$ voters: $V = \{v_1, \ldots, v_{2n}\}$, $3n$ candidates: $C = \{c_1, \ldots, c_{3n}\}$, and n winners. In addition, each voter may distribute b points among the candidates. We want the voters in V to cast their votes in a way that the following three conditions are satisfied:

1. For $j = 1, \ldots, n$, c_j has $b - w(a_j) + 1$ points.
2. For $j = n+1, \ldots, 2n$, c_j has at most b points.
3. For $j = 2n+1, \ldots, 3n$, c_j has exactly b points.

This can easily be done. Indeed, for $i = 1, \ldots, n$, voter v_i awards $b - w(a_i) + 1$ points to candidate c_i, and awards its remaining $w(a_i) - 1$ points to candidate c_{n+i}. Now, for $i = 1, \ldots, n$, voter $n + i$ awards all its b points to candidate $2n + i$.

We define the utility u of candidates as follows:

$$u(c_j) = \begin{cases} v(a_j) & j = 1, \ldots, n \\ 0 & j = n+1, \ldots, 3n \end{cases}$$

The transformation is clearly polynomial, so it only remains to verify that it is a reduction. Assume that there is a subset $A' \subseteq A$ with total weight at most b and total value at least t. Let $C = \{c_j : a_j \in A'\}$. The manipulator awards $w(a_j)$ points to each candidate $c \in C'$, raising the total score of these candidates to $b+1$. Since initially all candidates have at most b points, all candidates $c \in C'$ are among the n winners of the election. The total utility of these candidates is: $\sum_{c \in C'} u(c) = \sum_{a \in A'} v(a) \geq t$ (since for all $j = 1, \ldots, n$, $u(c_j) = v(a_j)$).

In the other direction, assume that the manipulator is able to distribute b points in a way that the winners of the election have total utility at least t. Recall that there are initially at least n candidates with b points and utility 0, and that ties are broken adversarially to the manipulator. Therefore, there must be a subset $C' \subseteq C$ of candidates that ultimately have a score of at least $b + 1$, such that their total utility is at least t. Let A' be the corresponding items in the KNAPSACK instance, i.e., $a_j \in A'$ iff $c_j \in C'$. The total weight of items in A' is at most b, as only b points were distributed among the candidates in C' by the manipulator, and each $c_j \in C'$ initially has $b - w(a_j) + 1$ points. It also holds that the total utility of the items in A' is exactly the total utility of the candidates in C', namely at least t. □

4 Control

Some voting protocols can be controlled by the authority conducting the election, which we refer to as the *chairman*, in the sense that the chairman can change the election's results. Some types of control available to the chairman are adding "spoiler" candidates, disqualifying candidates, registering new voters, or removing voters that were already registered. A study of these issues in the context of two well-known voting protocols was reported by Bartholdi, Tovey and Trick [1992], who found that control by adding and deleting candidates is $\mathcal{NP}$-hard even in the simple *Plurality*[4] protocol. Moreover, in most cases the complexity of deleting voters is identical to that of adding voters. Therefore, we focus hereinafter on control by adding voters.

The following formulation of the control (by adding voters) problem appeared in [Bartholdi *et al.*, 1992]: we are given a set C of candidates and a distinguished candidate $p \in C$; a set V of registered voters, and a set V' of voters that could register in time for the election. We are also given $r \in \mathbb{N}$, and have full knowledge of the voters' votes. We are asked whether it is possible to register at most r voters from V' in a way that makes p win the election.

As in the case of manipulation, we generalize this definition for our multi-winner setting:

Definition 3. In the CONTROL problem, we are given a set C of candidates, a set V of registered voters, a set V' of unregistered voters, the number of winners $k \in \mathbb{N}$, a utility function $u : C \rightarrow \mathbb{Z}$, the number of winners we are allowed to register $r \in \mathbb{N}$, and an integer $t \in \mathbb{N}$. We are asked whether it is possible to register at most r voters from V' such that in the resulting election, $\sum_{c \in W} u(c) \geq t$, where W is the set of winners, $|W| = k$.

Remark 2. Again, we assume that ties are broken adversarially to the chairman.

Proposition 3. CONTROL *in Bloc voting, Approval, and Cumulative voting is $\mathcal{NP}$-complete.*

Proof. By reduction from MAX k-COVER;[5] omitted due to lack of space. □

Proposition 4. CONTROL *in SNTV is in $\mathcal{P}$.*

Proof. We describe an algorithm, CONTROL-SNTV, that efficiently decides CONTROL in SNTV. Informally, the algorithm works as follows. It first calculates the number of points awarded to candidates by voters in V. Then, at each stage, the algorithm analyzes an election where the l top winners in the original election remain winners, and attempts to select the other $k - l$ winners in a way that maximizes utility. This is done by setting the *threshold* to be one point above the score of the $(l + 1)$-highest candidate; the algorithm *pushes* the scores of potential winners to this threshold.

A formal description of CONTROL-SNTV is given as Algorithm 1. The procedure PUSH works as follows: its first parameter is the threshold thr, and its second parameter is the number of candidates to be pushed, $pushNum$. The procedure also has implicit access to the input of CONTROL-SNTV, namely the parameters of the given CONTROL instance. PUSH returns a subset $V'' \subseteq V'$ to be registered. We say that the procedure *pushes* a candidate c to the threshold

[4]The Plurality protocol is identical to SNTV, when there is a single winner.

[5]See [Feige, 1998] for a definition and analysis of this problem.

Algorithm 1 Decides the CONTROL problem in SNTV.

procedure CONTROL-SNTV(C, V, V', k, u, r, t)
  $s[c] \leftarrow |\{v \in V : v$ votes for candidate $c\}|$
  Sort candidates by descending score ▷ Break ties by ascending utility
  Let the sorted candidates be $\{c_{i_1}, \ldots, c_{i_m}\}$
  for $l = 0, \ldots, k$ **do** ▷ Fix l top winners
    $V'' \leftarrow$PUSH$(s[c_{l+1}] + 1, k - l)$ ▷ Select other winners; see details below
    $u_l \leftarrow$ utility from election where V'' are registered
  end for
  if $\max_l u_l \geq t$ **then return true**
  else
    return false
  end if
end procedure

if exactly $thr - s[c]$ voters $v \in V'$ that vote for c are registered. In other words, the procedure registers enough voters from V' in order to ensure that c's score reaches the threshold. PUSH finds a subset C' of candidates of size at most $pushNum$ that maximizes $\sum_{c \in C} u(c)$, under the restriction that all candidates in C' can be simultaneously pushed to the threshold by registering a subset $V'' \subseteq V'$ s.t. $V'' \leq r$. The procedure returns this subset V''.

Now, assume we have a procedure PUSH that is always correct (in maximizing the utility of at most $k - l$ candidates it is able to push to the threshold $s[c_{l+1}] + 1$, while registering no more than r voters) and runs in polynomial time. Clearly, CONTROL-SNTV also runs in polynomial time. Furthermore:

Lemma 1. CONTROL-SNTV *correctly decides the* CONTROL *problem in SNTV.*

Proof. Let $W = \{c_{j_1}, \ldots, c_{j_k}\}$ be the k winners of the election that does not take into account the votes of voters in V' (the *original* election), sorted by descending score, and for candidates with identical score, by ascending utility. Let $W^* = \{c^*_{j_1}, \ldots, c^*_{j_k}\}$ be the candidates that won the controlled election with the highest utility, sorted by descending score, then by ascending utility; let $s^*[c]$ be the final score of candidate c in the optimal election. Let min be the smallest index such that $c_{j_{min}} \notin W^*$ (w.l.o.g. min exists). It holds that for all candidates $c \in W^*$, $s^*[c] \geq s[c_{j_{min}}]$. Now, we can assume w.l.o.g. that if $c \in W^*$ and $s^*[c] = s[c_{j_{min}}]$ then $c \in W$ (and consequently, $c = c_{j_q}$ for some $q < min$). Indeed, it must hold that $u[c] \leq u[c_{j_{min}}]$ (as tie-breaking is adversarial to the chairman), and if indeed $c \notin W$ even though $c \in W^*$, then the chairman must have registered voters that vote for c, although this can only lower the total utility.

It is sufficient to show that one of the elections that is considered by the algorithm has a set of winners with utility at least that of W^*. Indeed, let $W' = \{c_{j_1}, \ldots, c_{j_{min-1}}\} \subseteq W$; all other $k - min + 1$ candidates $c \in W^* \setminus W'$ have $s[c] \geq s[c_{j_{min}}] + 1$. The algorithm considers the election where the first $min - 1$ winners, namely W', remain fixed, and the threshold is $s[c_{j_{min}}] + 1$. Surely, it is possible to push all the candidates in $W^* \setminus W'$ to the threshold, and in such an election, the winners would be W^*. Since PUSH maximizes the utility of the $k - min + 1$ candidates it pushes to the threshold, the utility returned by PUSH for $l = min - 1$ is at least as large as the total utility of the winners in W^*. □

It remains to explain why the procedure PUSH can be implemented to run in polynomial time. Recall the KNAPSACK problem; a more general formulation of the problem is when there are two resource types. Each item has two weight measures, $w^1(a_i)$ and $w^2(a_i)$, and the knapsack has two capacities: b^1 and b^2. The requirement is that the total resources of the first type used do not exceed b^1, and the total resources of the second type do not exceed b^2. This problem, which often has more than two dimensions, is called MULTIDIMENSIONAL KNAPSACK. PUSH essentially solves a special case of the two-dimensional knapsack problem, where the capacities are $b^1 = r$ (the number of voters the chairman is allowed to register), and $b^2 = pushNum$ (the number of candidates to be pushed). If the threshold is thr, for each candidate c_j that is supported by at least $thr - s[c_j]$ voters in V', we set $w^1(a_j) = thr - s[c_j]$, $w^2(a_j) = 1$, and $v(a_j) = u(c_j)$. The MULTIDIMENSIONAL KNAPSACK problem can be solved in time that is polynomial in the number of items and the capacities of the knapsack [Kellerer *et al.*, 2004] (via dynamic programming, for example). Since in our case the capacities are bounded by m and $|V'|$, PUSH can be designed to run in polynomial time. □

5 Winner Determination

Some complex voting schemes are designed to be theoretically appealing in the sense that they satisfy some strict desiderata. Unfortunately, it might be the case that an attractive voting scheme is so complicated that even identifying the winners is an $\mathcal{NP}$-hard problem. This is a major issue, especially when one considers using such a protocol for real-life elections, as elections of this kind might always need to be resolved within a reasonable time frame.[6] Notice that in SNTV, Bloc Voting, Approval, and Cumulative Voting, it is clearly easy to tell who the winners are. In this section, however, we focus on the complex schemes introduced in Section 2.1.

Definition 4. In the WINNER-DETERMINATION problem, we are given the set of voters V, the set of candidates C, the number of winners $k \in \mathbb{N}$, misrepresentation values $\mu_{vc} \in \{0, 1, \ldots, m\}$, and $t \in \mathbb{N}$. We are asked whether there exists a subset $S \subseteq C$ such that $|S| = k$, with misrepresentation at most t.

Remark 3. Determining the set of winners is clearly harder than the above decision problem, as the set of winners *minimizes* misrepresentation.

[6]Thus the *negative* repercussions of a winner determination scheme being $\mathcal{NP}$-hard are even more pronounced than the *positive* repercussions of manipulation being $\mathcal{NP}$-hard; in the latter case a voting scheme might, unacceptably, still be susceptible to manipulation in the average case [Procaccia and Rosenschein, 2006].

Remark 4. For ease of exposition, we shall assume that n/k is an integer. This does not limit the generality of our results, as otherwise it is possible to pad the electorate with voters v such that $\mu_{vc} = 0$ for all $c \in C$.

Theorem 5. *The* WINNER-DETERMINATION *problem in Monroe's scheme and in the Chamberlin-Courant scheme is $\mathcal{NP}$-complete, even when the misrepresentation values are binary.*

Proof. By reduction from MAX k-COVER; omitted due to lack of space. □

Our hardness results relied on the implicit assumption that the number of winners k is not constant (in the previous sections as well). In the context of the WINNER-DETERMINATION problem, we are also interested in a setting where the number of winners is constant, as this is sometimes the case in real-life elections: the electorate grows, but the size of the parliament remains fixed.

Proposition 6. *When the number of winners satisfies $k = O(1)$, the* WINNER-DETERMINATION *problem in Monroe's scheme and in the Chamberlin-Courant scheme is in $\mathcal{P}$.*

Proof. Clearly the WINNER-DETERMINATION problem in the Chamberlin-Courant scheme can be solved efficiently when $k = O(1)$, as the size of the set $\mathcal{S}$, the set of subsets of candidates with size k, is polynomial in m. For a given $S \in \mathcal{S}$, finding the assignment f_S that minimizes misrepresentation in this scheme is simple: each voter v is assigned to $\text{argmin}_{c \in C} \mu_{vc}$.

In Monroe's scheme, by a similar consideration, it is sufficient to produce a procedure that efficiently computes the misrepresentation score of every $S \in \mathcal{S}$, i.e., finds a balanced assignment that minimizes misrepresentation in polynomial time.

We analyze a procedure that maintains at each stage a balanced assignment, and iteratively decreases misrepresentation.[7] Changes in the assignment are introduced by *cyclically right-shifting* (c.r.s.) sets of voters: each voter in a set $A = \{v_{i_1}, v_{i_2}, \ldots, v_{i_l}\}$ is shifted to the candidate that is assigned to its successor; the assignment remains balanced as the last voter is assigned to the first candidate. In more detail, if the current assignment is f_S, the algorithm singles out a set of voters $A = \{v_{i_1}, v_{i_2}, \ldots, v_{i_l}\}$, $l \leq k$, and modifies the assignment by defining the next assignment f'_S as follows:

$$f'_S(v_i) = \begin{cases} f_S(v_{i_{d+1 (\text{mod } l)}}) & v_i = v_{i_d} \in A \\ f_S(v_i) & v_i \notin A \end{cases} \qquad (1)$$

The procedure is formally described in Algorithm 2. It terminates after at most nm repetitions of the iterative step: at each iteration, the total misrepresentation decreases by at least 1, since the μ_{vc} are integers. On the other hand, the total misrepresentation cannot decrease below 0, and is initially at most $n \cdot \max_{v,c} \mu_{vc} \leq nm$. Moreover, the iterative step of the algorithm can be calculated efficiently: since k is constant, the number of possible cycles of length at most k is polynomial in n. We have that the complexity of WINNER-SELECTION in Monroe's scheme is polynomial — provided we are able to show that the procedure works!

[7] It is also possible to derive an efficient algorithm by applying bipartite matching algorithms to an appropriate graph, but the solution given here is self-contained.

Algorithm 2 Finds a balanced assignment that minimizes misrepresentation.

procedure ASSIGN(S)
  $f_S \leftarrow$ arbitrary assignment of n/k voters to each candidate in S
  loop
    if $\exists A \subseteq V\ s.t.\ |A| \leq k \wedge$ c.r.s. A strictly decreases misrepresentation **then**
      update f_S by performing the shift ▷ According to Equation (1)
    else
      return f_S
    end if
  end loop
end procedure

Lemma 2. ASSIGN *returns an optimal assignment.*

Proof. Consider a scenario where the procedure reaches the iterative step, but the current assignment is not optimal. We must show that the algorithm does not terminate at this point. Indeed, let $f^*_S : V \to S$ be a fixed optimal assignment. We consider the voters v such that $f_S(v) = f^*_S(v)$ to be *placed*, and the other voters to be *misplaced*. Assume without loss of generality that f^*_S minimizes the number of misplaced voters among all optimal assignments.

We claim that there is a set of $l \leq k$ voters that can be cyclically right-shifted in a way that places all l voters. Let v_{i_1} be a misplaced voter. In order to place it, it has to be assigned to the candidate $f^*_S(v_{i_1})$. Thus, one of the voters that f_S assigns to $f^*_S(v_{i_1})$ must be misplaced, otherwise f_S is not balanced; call this voter v_{i_2}. v_{i_2} can be placed by uprooting a voter v_{i_3} assigned to $f^*_S(v_{i_2})$. Iteratively repeating this line of reasoning, there must at some stage be a voter $v_{i_{d'}}$, $d' \leq k$, such that $f^*_S(v_{i_{d'}}) = f_S(v_{i_d})$ for some $d < d'$; this is true, since there are only k distinct candidates in S. Hence, the voters $\{v_{i_d}, v_{i_{d+1}}, \ldots, v_{i_{d'}}\}$ can be cyclically right-shifted in a way that places all $d' - d + 1 = l \leq k$ voters.

For any set of voters that can be placed by cyclic right-shifting, the shift must strictly decrease misrepresentation. Otherwise, by cyclically left-shifting the same set in f^*_S, we can obtain a new optimal and balanced assignment, in which more voters are placed compared to f^*_S; this is a contradiction to our assumption that f^*_S minimizes the number of misplaced voters.

It follows that there must be a set of at most k voters such that cyclically right-shifting the set strictly decreases the misrepresentation. Therefore, the procedure does not terminate prematurely. □

The proof of Proposition 6 is completed. □

In...	MANIPULATION	CONTROL
SNTV	$\mathcal{P}$	$\mathcal{P}$
Bloc	$\mathcal{P}$	$\mathcal{NP}$-c
Approval	$\mathcal{P}$	$\mathcal{NP}$-c
Cumulative	$\mathcal{NP}$-c	$\mathcal{NP}$-c

Table 1: The computational difficulty of Manipulation and Control in multi-winner protocols.

6 Conclusions

Table 1 summarizes the complexity of manipulation and control,[8] with respect to four protocols: SNTV, Bloc voting, Approval voting, and Cumulative voting. Of the four protocols, the only one that is computationally resistant to both manipulation and control is Cumulative voting. This protocol also has other advantages: it allows voters to express the intensities of their preferences, and encourages proportional results (albeit, without guaranteeing them). Therefore, Cumulative voting seems especially suitable as a method to aggregate agents' preferences.

One must remember in this context that $\mathcal{NP}$-hardness may not be a good enough guarantee of resistance to manipulation or control: an $\mathcal{NP}$-hard problem has an infinite number of hard instances, but it may have many more easy instances. Indeed, Procaccia and Rosenschein [2006] show that a specific family of voting protocols is susceptible to coalitional manipulation in the average-case, although the problem is hard in the worst-case. Nevertheless, $\mathcal{NP}$-hardness of manipulation or control should certainly be a *consideration* in favor of adopting some voting protocol.

While high complexity of manipulation or control in a voting scheme is interpreted positively, high complexity of winner determination is a major consideration against the scheme, and may in fact preclude its use in real-life settings. Winner determination is $\mathcal{NP}$-complete with respect to the theoretical voting schemes proposed by Monroe, and by Chamberlin and Courant. Monroe's scheme has received some attention in recent years. In particular, it has been shown that an election can be resolved with integer programming [Potthoff and Brams, 1998]. Unfortunately, solving an integer program is still difficult; this formulation does not even guarantee an efficient solution when the number of winners is constant. Such a solution is, however, given by Proposition 6. This implies that it is perhaps possible to use the scheme in settings where the size of the assembly is very small compared to the size of the electorate.

7 Acknowledgment

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

References

[Bartholdi and Orlin, 1991] J. Bartholdi and J. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8:341–354, 1991.

[Bartholdi *et al.*, 1989a] J. Bartholdi, C. A. Tovey, and M. A. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6:227–241, 1989.

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

[Bartholdi *et al.*, 1992] J. Bartholdi, C. A. Tovey, and M. A. Trick. How hard is it to control an election. *Mathematical and Computer Modelling*, 16:27–40, 1992.

[Brams and Fishburn, 2002] S. J. Brams and P. C. Fishburn. Voting procedures. In K. J. Arrow, A. K. Sen, and K. Suzumura, editors, *Handbook of Social Choice and Welfare*, chapter 4. North-Holland, 2002.

[Chamberlin and Courant, 1983] J. R. Chamberlin and P. N. Courant. Representative deliberations and representative decisions: Proportional representation and the Borda rule. *American Political Science Review*, 77(3):718–733, 1983.

[Conitzer and Sandholm, 2002] V. Conitzer and T. Sandholm. Complexity of manipulating elections with few candidates. In *Proceedings of the National Conference on Artificial Intelligence*, pages 314–319, Edmonton, Canada, July 2002.

[Conitzer *et al.*, 2003] V. Conitzer, J. Lang, and T. Sandholm. How many candidates are needed to make elections hard to manipulate? In *Proceedings of the International Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 201–214, Bloomington, Indiana, 2003.

[Ephrati and Rosenschein, 1997] Eithan Ephrati and Jeffrey S. Rosenschein. A heuristic technique for multiagent planning. *Annals of Mathematics and Artificial Intelligence*, 20:13–67, Spring 1997.

[Feige, 1998] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[Ghosh *et al.*, 1999] S. Ghosh, M. Mundhe, K. Hernandez, and S. Sen. Voting for movies: the anatomy of a recommender system. In *Proceedings of the Third Annual Conference on Autonomous Agents*, pages 434–435, 1999.

[Kellerer *et al.*, 2004] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 2004.

[Monroe, 1995] B. L. Monroe. Fully proportional representation. *American Political Science Review*, 89(4):925–940, 1995.

[Potthoff and Brams, 1998] R. F. Potthoff and S. J. Brams. Proportional representation: Broadening the options. *Journal of Theoretical Politics*, 10(2), 1998.

[Procaccia and Rosenschein, 2006] A. D. Procaccia and J. S. Rosenschein. Junta distributions and the average-case complexity of manipulating elections. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 497–504, 2006.

[8] Specifically, control by adding voters.