Journal of Artificial Intelligence Research 33 (2008) 149–178 Submitted 03/08; published 09/08

Complexity of Strategic Behavior in Multi-Winner Elections

Reshef Meir RESHEF24@CS.HUJI.AC.IL
Ariel D. Procaccia ARIELPRO@CS.HUJI.AC.IL
Jeffrey S. Rosenschein JEFF@CS.HUJI.AC.IL
Aviv Zohar AVIVZ@CS.HUJI.AC.IL
School of Engineering and Computer Science
The Hebrew University of Jerusalem

Abstract

Although recent years have seen a surge of interest in the computational aspects of social choice, no specific attention has previously been devoted to elections with multiple winners, e.g., elections of an assembly or committee. In this paper, we characterize the worst-case complexity of manipulation and control in the context of four prominent multi-winner voting systems, under different formulations of the strategic agent's goal.

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, Mundhe, Hernandez, and Sen (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, joint plans (Ephrati & Rosenschein, 1997), or schedules (Haynes, Sen, Arora, & Nadella, 1997).

Different aspects of voting rules have been explored by computer scientists. An issue that has been particularly well-studied is manipulation. In many settings, a voter may be better off revealing its preferences untruthfully. For instance, in real-life elections where each voter awards a single point to its favorite candidate, it may be judged pointless to vote for a candidate that appears, from polls, to be a sure loser, even if that candidate is a voter's truthful first choice.

The celebrated Gibbard-Satterthwaite Theorem (Gibbard, 1973; Satterthwaite, 1975) implies that under any non-dictatorial voting scheme (i.e., there is no single voter that always dictates the outcome of the election), there always exist elections in which a voter can improve its utility by lying about its true preferences.[2] 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. In other words, computational complexity may be an obstacle that prevents strategic behavior. The first to address this

1. We use the terms "voting schemes", "voting rules", "voting systems", and "voting protocols" interchangeably.
2. This theorem has also been generalized to the multiple winner setting (Duggan & Schwartz, 2000).

point were Bartholdi, Tovey and Trick (1989); Bartholdi and Orlin (1991) later showed that manipulating the important *Single Transferable Vote* (STV) voting rule is an $\mathcal{NP}$-complete problem.

More recently, it has been shown that voting protocols can be tweaked by adding an elimination preround, in a way that makes manipulation hard (Conitzer & Sandholm, 2003). Conitzer, Sandholm, and Lang (2007) 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 significant 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. The same rationale is at work here as well: if it is computationally hard to determine how to improve the outcome of the election by control, the chairman might give up on cheating altogether. Bartholdi, Tovey and Trick (1992) first analyzed the computational complexity of different methods of controlling an election in the *Plurality* and *Condorcet* protocols.

In this paper, we augment the classical problems of manipulation and control by introducing multiple winners. Specifically, we assume the manipulator has a utility function over the candidates, and that the manipulator's goal is to achieve a set of winners with total utility above some threshold. We study the abovementioned problems with respect to four simple but important multi-winner voting schemes: SNTV, Bloc voting, Approval, and Cumulative voting.

The paper proceeds as follows. In Section 2, we describe the voting rules in question. In Section 3, we deal with manipulation problems. In Section 4, we deal with control problems. We discuss related work in Section 5, and conclude in Section 6.

2. Multi-Winner Voting Schemes

In this section we present several multi-winner voting systems of significance. Although the discussion is self-contained, interested readers can find more details in an article by Brams and Fishburn (2002).

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

Multi-winner voting rules differ from single-winner ones in the properties that they are expected to satisfy. A major concern in multi-winner elections is *proportional representation*: a faction that consists of a fraction X of the population should be represented by approximately a fraction X of the seats in the assembly. This property is not satisfied by (generalizations of) many of the rules usually considered with respect to single-winner elections.

Thus, we here examine four of the prevalent multi-winner voting rules. 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.[3]

3. SNTV in single winner elections is also known as *Plurality*.

- *Bloc voting*: each voter gives one point to each of k candidates.[4]
- *Approval voting*: each voter approves or disapproves 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 preference, 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 & Fishburn, 2002).

Scoring rules are a prominent family of voting rules. A voting rule in this family is defined by a vector of integers $\vec{\alpha} = \langle \alpha_1, \ldots, \alpha_m \rangle$, where $\alpha_l \geq \alpha_{l+1}$ for $l = 1, \ldots, m-1$. Each voter reports a ranking of the candidates, thus awarding α_1 points to the top-ranked candidate, α_2 points to the second candidate, and in general α_l points to the candidate ranked in place l. Notice that SNTV is a family of scoring rules defined for each $|C|$ by the vector $\langle 1, 0, \ldots, 0 \rangle$, and Bloc is the family of scoring rules defined for each $|C|, k$ by the vector $\langle 1, \ldots, 1, 0, \ldots, 0 \rangle$, where the number of 1's is k.

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, since it may lead to a socially undesirable outcome emerging as the winner of the election. Therefore, computational resistance to manipulation is considered an advantage.

In the classical formalization of the manipulation problem (Bartholdi et al., 1989), 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. A very general formulation is given by the following definition.

Definition 3.1. 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, $|W| = k$.

Notice that the number of winners k is a parameter of the problem.

Remark 3.2. The manipulator's utility function is implicitly assumed to be additive. One can consider more elaborate utility functions, such as the ones investigated in the context of combinatorial auctions, but that is beyond the scope of this paper.

4. Bloc voting is also known as k-Approval.

Remark 3.3. We make the standard assumption that tie-breaking is adversarial to the manipulator (Conitzer et al., 2007), i.e., if there are several candidates that perform equally well in the election, the ones with the lower utility for the manipulator will be elected. When the number of winners is $k = 1$, this assumption is equivalent to formulating the manipulation problems in their *unique winner* version, where the ballot must be cast in a way that some designated candidate is strictly better than the rest (e.g., has higher score).

One might argue that the general formulation of the problem given above makes manipulation harder. Indeed, the manipulator might have the following, more specific, goals in mind.

1. The manipulator has a specific candidate whom he is interested in seeing among the winners (constructive manipulation).

2. The manipulator has a specific candidate whom he is interested in excluding from the set of winners (destructive manipulation) (Conitzer et al., 2007).

3. The manipulator has some (additive) boolean-valued utility function over the candidates $u : C \to \{0, 1\}$.

The first and second settings are naturally special cases of the third, while the third is a special case of Definition 3. We intend to explore all the foregoing formulations of the Manipulation problem. Notice that one can also consider other goals, for example when the manipulator has a favorite set of candidates and he is interested in seeing *all of them*, or *as many as possible of them*, among the winners. However, we will only investigate these goals insofar as they are special cases of a boolean-valued utility function.

Remark 3.4. Unless explicitly mentioned otherwise, we usually assume a general (additive) utility function, as in Definition 3.1.

We will find it convenient to represent SNTV and Bloc voting using a common framework. We consider *l-Bloc voting* rules—voting rules where every voter gives one point to each of exactly l candidates, where $l \leq k$. Notice that in SNTV $l = 1$, while in Bloc voting $l = k$. We remind the reader that the number of winners k is not constant, but is rather a parameter of the Manipulation problem.

Proposition 3.5. *Let $l \in \{1, \ldots, k\}$. Then* Manipulation *in l-Bloc voting is in $\mathcal{P}$.*

Proof. The manipulator is faced with the score awarded to the candidates by the voters in V; let $s[c]$ be the total score of candidate c. Order the candidates by their score, and let s_0 be the score of the k'th highest candidate. For example, if $k = 3$, $|C| = m = 4$, and the initial scores are 8, 5, 5, 3, then $s_0 = 5$. In addition, let

$$A = \{c \in C : \; s[c] > s_0\}.$$

Notice that $|A| \leq k - 1$. Let

$$B = \{c \in C : \; s[c] = s_0\}.$$

B may be large, and in particular if all the candidates have the same score, then $B = C$.

Now, candidates in A with at least $s_0 + 2$ initial points will be elected regardless of the actions of the manipulator, as the manipulator can award at most one point to any candidate. Candidates in A with exactly s_0+1 points will be elected, unless other candidates with lower utility ultimately receive $s_0 + 1$ points due to the manipulator's vote—as ties are broken adversarially to the manipulator. Let us now examine the candidates with less than s_0 points. Notice that candidates with $s[c] \leq s_0 - 2$ will lose in any case. Moreover, since ties are broken adversarially, voting for candidates with $s_0 - 1$ points cannot benefit the manipulator. To conclude the point, the manipulator can do no better than to make sure the candidates in B (with exactly s_0 points) that are eventually elected have as high utility as possible.

Therefore, to put it simply, the manipulator's optimal strategy is to vote for the top (in terms of utility) $k - |A|$ (top l if $l < k - |A|$) candidates in B, thus guaranteeing that these candidates be among the winners, and cast its remaining votes in favor of some candidates in A (which will win anyway). This discussion leads to the conclusion that Algorithm 1 decides the Manipulation problem. Clearly the computational complexity of

Algorithm 1 Decides the Manipulation problem in l-Bloc voting

procedure Manipulate-Bloc(V, C, k, u, t, l)
  $s[c] \leftarrow |\{v \in V : v \text{ votes for candidate } c\}|$
  $s_0 \leftarrow$ score of k'th highest candidate
  $A \leftarrow \{c \in C : s[c] > s_0\}$ ▷ $|A| \leq k - 1$
  $B \leftarrow \{c \in C : s[c] = s_0\}$, ordered by $u(c)$ in decreasing order
  if $l \leq k - |A|$ **then**
    manipulator votes for top l candidates in B
  else
    manipulator votes for top $k - |A|$ candidates in B and $l + |A| - k$ candidates in A
  end if
  if utility of winners $\geq t$ **then**
    return true
  else
    return false
  end if
end procedure

the algorithm is polynomial in the input size. □

We have the following immediate corollary:

Corollary 3.6. Manipulation *in SNTV and Bloc voting is in $\mathcal{P}$.*

The situation in Approval voting is not dissimilar. Indeed, the question is: can a voter gain by approving more than k candidates? *A priori*, the answer is yes. However, given the votes of all other voters, clearly the manipulator cannot gain by approving candidates other than the k eventual winners of the election. On the other hand, the manipulator also does not benefit from approving *less* than k voters. Say the manipulator approved $l < k$

voters, and the candidates in W, where $|W| = k$, eventually won the election. Let C^* be the candidates that the manipulator approved, and $W^* = W \setminus C^*$ be the winners that the manipulator did not approve; $W^* \geq k - l$. If the manipulator approves the l candidates in C^* as well as some $k - l$ candidates in W^*, the set of winners is clearly still going to be W. Therefore, the manipulator can do no better than approve exactly k candidates; we have already demonstrated in Proposition 3.5 that this can be accomplished optimally and efficiently. Therefore:

Corollary 3.7. Manipulation *in Approval is in $\mathcal{P}$.*

Remark 3.8. Formally, manipulation in Approval is a subtle issue, since the issue may be ill-defined when the voters are assumed to have linear preferences over the candidates. In this case, there are multiple sincere ballots (where all approved candidates are preferred to all disapproved candidates). In some specific settings, a voter cannot gain by casting an insincere ballot (Endriss, 2007), but this is not always true. In any case, the manipulation problem according to our definition is well-defined and nontrivial in Approval.

In contrast to the abovementioned three voting rules, Cumulative voting turns out to be computationally hard to manipulate under a general utility function.

Proposition 3.9. Manipulation *in Cumulative voting is $\mathcal{NP}$-complete.*

The implicit assumption made in the proposition is that the number of points to be distributed is not a constant, but rather a parameter of the Manipulation problem under Cumulative voting.

The proof of Proposition 3.9 relies on a reduction from one of the most well-known $\mathcal{NP}$-complete problems, the Knapsack problem.

Definition 3.10. In the Knapsack problem, we are given a set of items $A = \{a_1, \ldots, a_n\}$, for each $a \in A$ a weight $w(a) \in \mathbb{N}$ and a value $\upsilon(a)$, a capacity $b \in \mathbb{N}$, and $t \in \mathbb{N}$. We are asked whether there is a subset $A' \subseteq A$ such that $\sum_{a \in A'} \upsilon(a) \geq t$ while $\sum_{a \in A'} w(a) \leq b$.

Proof of Proposition 3.9. 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 $\langle A, w, \upsilon, b, t \rangle$ 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 c_{2n+i}.

We define the utility u of candidates as follows:

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

The transformation is clearly polynomial time computable, 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: $\sum_{c \in C'} u(c) = \sum_{a \in A'} \upsilon(a) \geq t$ (since for all $j = 1, \ldots, n$, $u(c_j) = \upsilon(a_j)$).

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., $a_j \in A'$ if and only if $c_j \in C'$. 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. □

The next proposition gives a negative answer to the question of whether Manipulation in Cumulative voting is still hard under more restricted formulations of the manipulator's goal, as discussed at the beginning of the section. Indeed, we put forward an algorithm that decides the problem under any boolean-valued utility function.

Proposition 3.11. Manipulation *in Cumulative voting with any boolean-valued utility function* $u : C \to \{0, 1\}$ *is in* $\mathcal{P}$.

Remark 3.12. The result holds even if the number of points to be distributed is exponential in the number of voters and candidates.

Proof of Proposition 3.11. Let $s[c]$ be the score of candidate $c \in C$ before the manipulator has cast his vote, and $s^*[c]$ be c's score when the manipulator's vote is taken into account. Assume without loss of generality that $s[c_1] \geq s[c_2] \geq \ldots \geq s[c_m]$. Let $D = \{d_1, d_2, \ldots\}$ be the set of desirable candidates $d \in C$ with $u(d) = 1$, and again assume these are sorted by nonincreasing scores.

Informally, we are going to find a threshold *thresh* such that pushing t candidates above the threshold guarantees their victory. Then we will check whether it is possible to distribute L points such that at least t candidates pass this threshold, where L is the number of points available to each voter.

Formally, consider Algorithm 2 (w.l.o.g. $k \geq t$ and $|D| \geq t$, otherwise manipulation is impossible). The algorithm clearly halts in polynomial time. It only remains to prove the correctness of the algorithm.

Algorithm 2 Decides Manipulation in Cumulative voting with boolean-valued utility

1: $j^* \leftarrow max\{j: \ |\{c_1, c_2, \ldots, c_{j-1}\} \cap D| + k + 1 - j \geq t$ and $c_j \notin D\}$ ▷ j^* exists, since the condition holds for the first candidate not in D
2: $thresh \leftarrow s[c_{j^*}]$
3: $S \leftarrow \sum_{j=1}^{t} \max\{0, thresh + 1 - s[d_j]\}$
4: **if** $S \leq L$ **then**
5: **return true**
6: **else**
7: **return false**
8: **end if**

Lemma 3.13. *Algorithm 2 correctly decides* Manipulation *in Cumulative voting with any boolean-valued utility function.*

Proof. Denote by $\hat{W} = \{c_1, \ldots, c_k\}$ the k candidates with highest score (sorted) before the manipulator's vote, and by W the final set of k winners. The threshold candidate c_{j^*} partitions $\hat{W}$ into two disjoint subsets: $\hat{W}_u = \{c_1, \ldots, c_{j^*-1}\}, \hat{W}_d = \{c_{j^*}, \ldots, c_k\}$. By the maximality of j^*, it holds that:

$$|\hat{W}_u \cap D| + |\hat{W}_d| = |\hat{W}_u \cap D| + (k + 1 - j^*) = t. \tag{1}$$

Note that S is the exact number of votes required to push t desirable candidates above the threshold. Now, we must show that the manipulator can cast his vote in a way such that the winner set W satisfies $|W \cap D| \geq t$ if, and only if, $L \geq S$.

Suppose first that $S \leq L$. Then it is clearly possible to push t desirable candidates above *thresh*. $\hat{W}_u \cap D$ were above the threshold already; it follows that $\hat{W}_d$ was replaced entirely by desirable candidates.

Let $W = \{w_1, \ldots, w_k\}$ be the set of *new* winners. In particular, we can write $W = \hat{W}_u \cup \{w_{j^*}, \ldots, w_k\}$. $\hat{W}_u$ contains $|\hat{W}_u \cap D|$ desirable candidates, while $\{w_{j^*}, \ldots, w_k\}$ consists purely of desirable candidates. By Equation (1):

$$\begin{aligned} |W \cap D| &= |\hat{W}_u \cap D| + |\{w_{j^*}, \ldots, w_k\}| \\ &= |\hat{W}_u \cap D| + |\hat{W}_d| \\ &= t \end{aligned}$$

Conversely, suppose $S > L$. We must show that the manipulator cannot distribute L points in a way such that t candidates from D are among the winners.

Clearly there is no possibility to push t desirable candidates above *thresh*. Consider some ballot cast by the manipulator, and assume w.l.o.g. that the manipulator distributed points only among the candidates in D. Denote the new set of winners by $W = W_u \uplus W_d$, where

$$W_u = \{c \in C: \ s^*[c] > thresh\}$$
$$W_d = \{c \in C: \ s^*[c] \leq thresh\}.$$

We claim that

$$|W_u \cap \overline{D}| = k - t, \tag{2}$$

where $\overline{D} = C \setminus D$. Indeed, by Equation (1)

$$|\hat{W}_u \cap D| = t - k - 1 + j^*,$$

and therefore

$$\begin{aligned} |W_u \cap \overline{D}| = |\hat{W}_u \cap \overline{D}| &= |\hat{W}_u| - |\hat{W}_u \cap D| \\ &= (j^* - 1) - (t - k - 1 + j^*) \\ &= k - t \end{aligned}$$

The first equality follows from the fact that no points were distributed to the candidates in $\overline{D}$.

Denote by F the set of candidates that were pushed above the threshold. Formally:

$$F = \{c \in D : s^*[c] > \mathit{thresh} \text{ and } s[c] \leq \mathit{thresh}\}$$

Thus:

$$W_u = \hat{W}_u \uplus F.$$

Let w^* be the new position of candidate c_{j^*} when the candidates are sorted by nonincreasing $s^*[c]$. It holds that

$$w^* = j^* + |F|.$$

We now claim that

$$|W_d \cap \overline{D}| \geq 1. \tag{3}$$

Indeed,

$$\begin{array}{cc} |W_u \cap D| < t & \Rightarrow \\ |\hat{W}_u \cap D| + |F| = |W_u \cap D| < t = |\hat{W}_u \cap D| + k + 1 - j^* & \Rightarrow \\ |F| < k + 1 - j^* & \Rightarrow \\ w^* = j^* + |F| < j^* + k + 1 - j^* = k + 1 & \Rightarrow \\ w^* \leq k & \Rightarrow \\ c_{j^*} \in W_d & \Rightarrow \\ |W_d \cap \overline{D}| \geq 1 & \end{array}$$

By combining Equations (2) and (3), we finally obtain:

$$\begin{aligned} |W \cap D| &= k - |W \cap \overline{D}| \\ &= k - (|W_u \cap \overline{D}| + |W_d \cap \overline{D}|) \\ &\leq k - (k - t + 1) \\ &= t - 1 \\ &< t \end{aligned}$$

□

The proof of Proposition 3.11 is completed. □

Remark 3.14. The proof shows that the manipulation of Cumulative voting by a *coalition* of (even weighted) voters, as in the work of Conitzer et al. (2007), is tractable under a boolean-valued utility function. This follows by simply joining the (weighted) score pools of all the voters in the coalition.

Remark 3.15. It is possible to show that if the number of points to be distributed is polynomially bounded, manipulation of Cumulative voting is in $\mathcal{P}$ even under general utility functions.

4. Control

In the control setting, we assume that the authority controlling the election (hereinafter, the *chairman*) has the power to tweak the election's electorate or slate of candidates in a way that might change the outcome. This is also a form of undesirable strategic behavior, but on the part of a behind-the-scenes player who is not supposed to take an active part in the election.

In one setting, the chairman might add or remove voters that support his candidate, but the number of voters he can add/remove without alerting attention to his actions is limited. The problems are formally defined as follows:

Definition 4.1. *In the problem of* Control by Adding Voters, *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 $u : C \to \mathbb{Z}$, and integers $r, t \in \mathbb{N}$. 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, $|W| = k$.*

Definition 4.2. *In the problem of* Control by Removing Voters, *we are given a set C of candidates, a set V of registered voters, the number of winners $k \in \mathbb{N}$, a utility function $u : C \to \mathbb{Z}$, and integers $r, t \in \mathbb{N}$. We are asked whether it is possible to remove 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, $|W| = k$.*

Another possible misuse of the chairman's authority is tampering with the slate of candidates. Removing candidates is obviously helpful, but even adding candidates can sometimes tip the scales in the direction of the chairman's favorites.

Definition 4.3. *In the problem of* Control by Adding Candidates, *we are given a set C of registered candidates, a set C' of unregistered candidates, a set V of voters, the number of winners k, a utility function $u : C \cup C' \to \mathbb{Z}$, and integers $r, t \in \mathbb{N}$. All voters have preferences over all candidates $C \cup C'$. We are asked whether it is possible to add at most r candidates C'' from C', such that in the resulting elections on $C \cup C''$, $\sum_{c \in W} u(c) \geq t$, where W is the set of winners, $|W| = k$.*

Definition 4.4. *In the problem of* Control by Removing Candidates, *we are given a set C of candidates, a set V of voters, the number of winners k, a utility function $u : C \to \mathbb{Z}$, and integers $r, t \in \mathbb{N}$. We are asked whether it is possible to remove at most r candidates from C', such that in the resulting elections $\sum_{c \in W} u(c) \geq t$, where W is the set of winners, $|W| = k$.*

Some clarification is in order. In the context of scoring rules, the assumption in the above two problems is that the voters have rankings of all the candidates in $C \cup C'$. Therefore, if some candidates are added or removed, the voters' preferences over the new set of candidates are still well-defined. The same goes for Approval: each voter approves or disapproves every candidates in $C \cup C'$. However, in the context of Cumulative voting, the problems of control by adding/removing candidates are *not* well-defined. Indeed, one would require a specification of how the voters distribute their points among every possible subset of candidates, and this would require a representation of exponential size.[5] Consequently, we do not consider control by adding or removing candidates in Cumulative voting.

Remark 4.5. Some authors (e.g., Hemaspaandra et al., 2007b) have considered other types of control, such as control by partitioning the set of voters. In this paper, we will restrict our attention to the four types of control mentioned above.

Remark 4.6. Unless stated otherwise, we assume that ties are broken adversarially to the chairman.

As before, unless explicitly mentioned otherwise, we are going to assume a general additive utility function, as in the above definitions.

Remark 4.7. It is clear that all the above computational problems are in $\mathcal{NP}$ for all voting rules in question. Therefore, in our $\mathcal{NP}$-completeness proofs we will only show hardness.

4.1 Controlling the Set of Voters

Control by Adding Voters is tractable in SNTV, though the procedure is not trivial. Bartholdi et al. (1992) showed that control by adding voters is easy in SNTV when there is a single winner, so the former result can be seen as an extension of the latter (to the case where there are multiple winners and a general [additive] utility function).

Proposition 4.8. Control by Adding Voters *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 (see Figure 1 for an illustration).

A formal description of Control-SNTV is given as Algorithm 3. 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 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

5. It is possible to imagine compact representations, but that is beyond the scope of this paper.

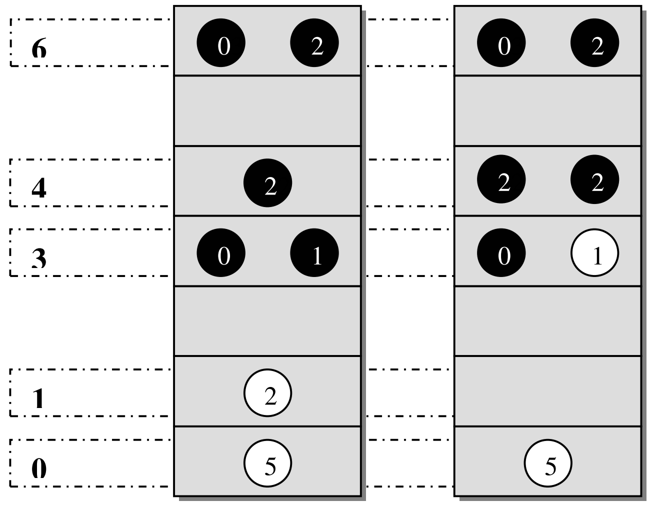

Figure 1: The left panel illustrates an input of the CONTROL problem in SNTV. Each candidate is represented by a circled number—the utility of the candidate. The location of the circle determines the score of the candidate, based on the voters in V. Let $k = 5$; the winners are blackened. Now, assume that there are 6 voters in V', 3 voting for each of the two bottom candidates, and that $r = 3$. The chairman can award 3 points to the candidate with utility 5 and score 0, but that would not change the result of the election. Alternatively, the chairman can award 3 points to candidate with utility 2 and score 1, thus improving the utility by 1, as can be seen in the right panel. This election is considered by the algorithm when $l = 4$, $s[c_{i_5}] = 3$, and the threshold is 4.

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 4.9. 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 some controlled election with maximum 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, otherwise $W = W^*$ and we are done). It holds that for all candidates $c \in W^*$, $s^*[c] \geq s[c_{j_{min}}]$. Now,

Algorithm 3 Decides the Control problem in SNTV.

procedure Control-SNTV(C, V, V', k, u, r, t)
    $s[c] \leftarrow |\{v \in V :\ v \text{ votes for candidate } c\}|$
    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

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 or keep it unchanged.

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 $w^2(a_i)$, that specify how much resource it consumes from each type, and the knapsack has two capacities: b^1 and b^2. The requirement is that the total amount of resource of the first type that is consumed does not exceed b^1, and the total use of resource of the second type does 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, Pferschy, & Pisinger, 2004) (via dynamic programming, for example). Since in our case the capacities are bounded by $|V'|$ and m, PUSH can be designed to run in polynomial time. □

The following lemma allows us to extend our results for CONTROL BY ADDING VOTERS to CONTROL BY REMOVING VOTERS, and vice versa. We shall momentarily apply it to SNTV, but it will also prove useful later on.

Lemma 4.10. *Let $\mathcal{R} = \langle V, C, r, t, k, u\rangle$ be an instance of* CONTROL BY REMOVING VOTERS *in some voting rule, and let $\mathcal{A} = \langle V', V'', C^*, r^*, t^*, k^*, u^*\rangle$ be an instance of* CONTROL BY ADDING VOTERS *in the same voting rule, such that:*

$$\begin{aligned} C^* &= C \\ r^* &= r \\ t^* &= t - \sum_{c\in C} u(c) \\ k^* &= |C| - k \\ u^*(c) &= -u(c) \\ V'' &= V \end{aligned}$$

Let U be the subset of voters selected by the chairman from $V = V''$. Denote by $s_U[c], s^*_U[c]$ the final score of candidate c in the election obtained by removing or adding the voters in U, respectively. If it holds that

$$\forall c, c' \in C, \forall U \subseteq V, \quad s^*_U[c] \geq s^*_U[c'] \;\Leftrightarrow\; s_U[c] \leq s_U[c'] \tag{4}$$

then $\mathcal{R}$ is in CONTROL BY REMOVING VOTERS *if and only if $\mathcal{A}$ is in* CONTROL BY ADDING VOTERS.

Proof. From the condition (4) on s_U, s^*_U it holds that the $k^* = |C| - k$ winners of the constructed instance are exactly the $|C| - k$ losers of the original instance.[6] That is, if W are the winners in the given instance and W^* are the winners in the constructed instance, we have that $W^* = C \setminus W$.

From this it follows that,

$$\begin{aligned} \sum_{c\in W} u(c) - t &= \sum_{c\in C\setminus W^*} u(c) - (t^* + \sum_{c\in C} u(c)) \\ &= -\sum_{c\in C\setminus W^*} u^*(c) + \sum_{c\in C} u^*(c) - t^* \\ &= \sum_{c\in W^*} u^*(c) - t^* \end{aligned}$$

Thus, for any choice of subset U to be removed or added, $\sum_{c\in W} u(c) \geq t$ if, and only if, $\sum_{c\in W^*} u^*(c) \geq t^*$. We conclude that the given instance is a “yes” instance if and only if the constructed instance is a “yes” instance. □

6. This statement also takes into account the tie-breaking scheme, as candidates with lower utility in the original instance are now candidates with higher utility.

Applying the lemma, we easily obtain:

Proposition 4.11. Control by Removing Voters *in SNTV is in* $\mathcal{P}$.

Proof. We will give a polynomial time reduction from Control by Removing Voters in SNTV to Control by Adding Voters in SNTV, which was shown above (Proposition 4.8) to be in $\mathcal{P}$. Given an instance $\langle V, C, r, t, k, u\rangle$ of Control by Removing Voters, define an equivalent instance $\langle V', V'', C^*, r^*, t^*, k^*, u^*\rangle$ of the latter problem. We want to use Lemma 4.10, so we need to define V' correctly.

For each voter $v \in V = V''$, let $f(v) \in C$ be the candidate that voter v ranks first; $f(v)$ gives all the relevant information about voter v's ballot. The ballots of the voters in V' are defined by the following rule. For each candidate $c \in C$,

$$|\{v' \in V' : f(v') = c\}| = |V| - |\{v \in V : f(v) = c\}|$$

It holds that:

$$s_U[c] = |\{v \in V : f(v) = c\}| - |\{v \in U : f(v) = c\}|$$

From the definition of V',

$$\begin{aligned} s^*_U[c] &= |\{v' \in V' : f(v') = c\}| + |\{v' \in U : f(v') = c\}| \\ &= |V| - |\{v \in V : f(v) = c\}| + |\{v \in U : f(v) = c\}| \\ &= |V| - (|\{v \in V : f(v) = c\}| - |\{v \in U : f(v) = c\}|) \\ &= |V| - s_U[c] \end{aligned}$$

Hence for all $c, c' \in C$, and for any $U \subseteq V$:

$$s^*_U[c] \geq s^*_U[c'] \;\Leftrightarrow\; s_U[c] \leq s_U[c'].$$

This implies that the conditions of Lemma 4.10 hold, thus there is a polynomial reduction from Control by Removing Voters in SNTV to Control by Adding Voters in SNTV. Since the latter problem is in $\mathcal{P}$, so is the former. ☐

In the rest of the section we prove that control by adding/removing voters is hard in the other three voting rules under consideration, even if the chairman simply wants to include or exclude a specific candidate. In fact, this statement includes 12 different results (3 voting rules × adding/removing × include/exclude). Instead of proving each result separately, we will use some generic reductions. Our proof scheme is as follows: we shall first establish that Control by Removing Voters is hard in Approval, Bloc, and Cumulative voting, even if the chairman wants to include a candidate. We will then use Lemma 4.10 to show that adding voters is hard in the foregoing rules, even if the chairman wants to exclude a candidate. Finally, Lemma 4.17 will give us the six remaining results: removing while excluding, and adding while including. The overall scheme is illustrated in Figure 2.

The following result is known from the work of Hemaspaandra et al. (2007b).

Proposition 4.12. *(Hemaspaandra et al., 2007b)* Control by Removing voters *in Approval is* $\mathcal{NP}$*-complete, even if the chairman is trying to make a distinguished candidate win in single winner elections.*

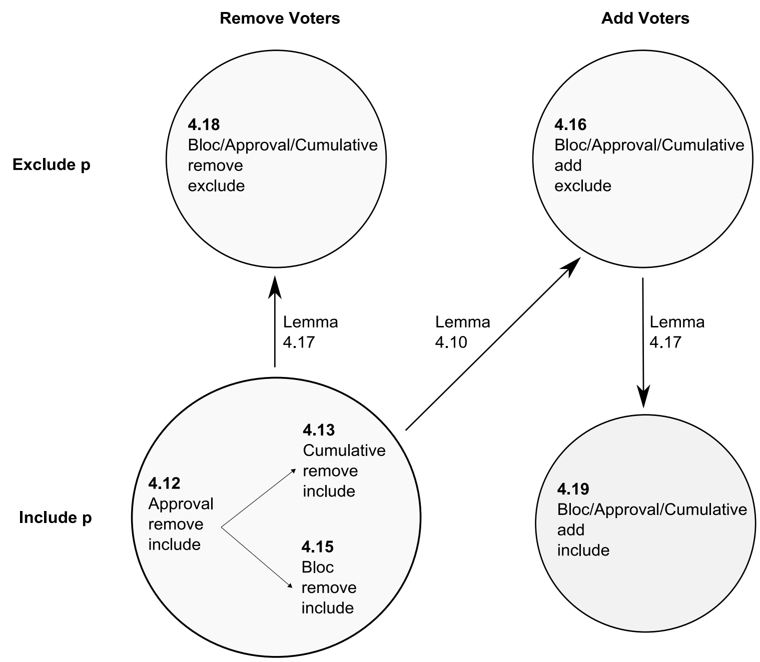

Figure 2: Scheme of the hardness proofs of control by adding/removing voters when the chairman wants to include/exclude a distinguished candidate, in the voting rules: Bloc, Approval, Cumulative voting.

This result, as with some other results that appear later, is stated for single winner elections, i.e., when the number of winners k satisfies $k = 1$. Clearly the hardness of the problem when $k = 1$ implies the hardness of the more general formulation of the problem when k is also a parameter.

Proposition 4.13. Control by Removing Voters *in Cumulative voting is* $\mathcal{NP}$*-complete, even if the chairman is trying to make a distinguished candidate win in single winner elections.*

Remark 4.14. In instances of control problems where the chairman simply wants to include/exclude a distinguished candidate, we replace the utility function u with the distinguished candidate p, i.e., $u(p) = 1$ (respectively, $u(p) = -1$) for include (resp., for exclude) and $u(c) = 0$ for all other candidates c. The threshold is $t = 1$ (resp., $t = 0$).

Proof of Proposition 4.13. We will use a reduction from Control by Removing Voters in Approval. Let $\langle V, C, p, r\rangle$ be an instance of this problem in Approval, where $p \in C$ is the distinguished candidate (and $k = 1$). Define an instance of the above problem in Cumulative voting as follows: The pool of points satisfies $L^* = |C|$; $r^* = r$; $p^* = p$; $C^* = C \uplus C'$, where C' contains $|V| \cdot L^*$ candidates.

We now construct a set of voters. For each voter $v \in V$, we add to V' a voter that awards 1 point to every candidate whom v approves, and then gives all other points (no more than L^*) to distinct candidates in C'. V'' contains two more voters; each of these voters gives one point to each candidate in C. Finally, let $V^* = V' \uplus V''$.

Removing voters from V'' cannot promote p^*, so the chairman may limit himself without loss of generality to removing voters from V'. Now, every candidate in C' has at most one point, so none of them beats any candidate from C (each of whom has at least two points). Furthermore, for every selection of voters from V' and corresponding voters from V to remove, each candidate in C gets the same score in the new instance as he got in the original instance, plus 2. This directly implies that control is possible in the constructed instance (of Cumulative voting) if and only if it is possible in the given instance (of Approval). □

In order to complete our hardness proofs for removing voters while including a distinguished candidate, we give a similar result for Bloc. Notice, however, that here the reduction constructs instances with multiple winners.

Proposition 4.15. CONTROL BY REMOVING VOTERS *in Bloc is $\mathcal{NP}$-complete, even when the chairman simply wants to include a distinguished candidate among the winners.*

Proof. We prove the proposition by a polynomial time reduction from CONTROL BY REMOVING VOTERS in Approval. Let $\langle V, C, p, k, r\rangle$ be an instance of the latter problem (with k winners; it is possible to let $k = 1$). Denote, as usual, $n = |V|$ and $m = |C|$. Define: $C^* = C \uplus C' \uplus C''$, where C' contains m new candidates, and C'' contains $10 \cdot k \cdot n$ new candidates; $p^* = p$; $k^* = k + m$; $r^* = r$. We need to design the new instance such that all the candidates in C' will be among winners, and all the candidates in C'' to be among the losers. We define the a voter set accordingly.

1. V' contains one voter for each voter in V (n in total). For each $v \in V$, v' gives a point to all candidates in C that v approved. His other points go to candidates in C', and then to C''.

2. V'' contains $r + 2$ voters. Each voter gives a point to each candidate in C, and gives his other k points to candidates in C' (arbitrarily).

3. V''' contains $4n$ voters. Each voter awards points to all C', and to k more candidates from C'' (in a way that will be specified later).

Finally let $V^* = V' \uplus V'' \uplus V'''$.

We claim that $\langle V^*, C^*, p^*, k^*, r^*\rangle \in$ CONTROL BY REMOVING VOTERS in Bloc if and only if $\langle V, C, p, k, r\rangle \in$ CONTROL BY REMOVING VOTERS in Approval.

For each candidate c, denote $s[c], s^*[c]$ the total number of votes c obtains in original and constructed instances, *before* removing any voters. Note that the whole set C'' gets at most $(4n)k + nk$ votes, which is less than its size ($10kn$), so it is possible to scatter the votes of V'' such that each candidate in C'' has at most 1 point. In addition, each candidate in C' has at least $4n$ points. It also holds that

$$\forall c \in C,\ s^*[c] = s[c] + r + 2,$$

(from the votes of V', V''), and thus:

$$\forall c \in C,\ 4n - r > r + n + 2 \geq s^*[c] > r + 1.$$

We can now conclude that:

$$\forall c \in C, c' \in C',\ s^*[c] < s^*[c'] - r,$$

and

$$\forall c \in C, c'' \in C'',\ s^*[c] > s^*[c''] + r.$$

Without loss of generality the chairman removes only voters from $V'(=V)$, since only these votes may influence the result of the elections. Denote by $s_U[c], s_U^*[c]$ the score of candidate c in both instances, *after* removing the subset $U \subseteq V = V'$. For any U such that $|U| \leq r$ it holds that:

$$\forall c_1, c_2 \in C,\ s_U^*[c_1] \geq s_U^*[c_2] \iff s_U[c_1] \geq s_U[c_2].$$

Moreover,

$$\forall c \in C, c' \in C',\ s_U^*[c] < s_U^*[c'],$$

and

$$\forall c \in C, c'' \in C'',\ s_U^*[c] > s_U^*[c''].$$

In other words, all the candidates in C' will be among winners, whereas all the candidates in C'' will be among losers, and the ranking of the candidates in C will not change.

We conclude that the k^* winners of the constructed instance are exactly the m candidates of C' together with the k winners of the given Approval instance. Thus control is possible in the original instance of Approval (i.e., it is possible to make p win) if and only if it is possible in the constructed instance of Bloc. □

As promised, we now use Lemma 4.10 to transform the results of all three voting rules from *Control (Include winner) by Removing Voters* to *Control (Exclude winner) by Adding Voters.*

Proposition 4.16. Control by Adding Voters *in Approval, Bloc, and Cumulative voting is $\mathcal{NP}$-hard, even when the chairman simply wants to exclude a distinguished candidate from the set of winners.*

Proof. We prove the proposition using Lemma 4.10 and with similar notations. Consider an instance of Control by Removing Voters when the chairman wants to include a candidate. Then the utility function u is 1 on p and 0 otherwise, while $t = 1$. The utility function u^* constructed by Lemma 4.10 is -1 on p and 0 otherwise, and the threshold t^* is $t - 1 = 0$; that is, the chairman wants to exclude p. Now we can obtain the desired result directly by showing that the lemma's condition holds. In other words, we must show that for the three voting rules in question, given an instance of Control by Removing Voters, it is possible to construct an instance of Control by Adding Voters that satisfies the condition (4).

Approval and Bloc: The proof here is very similar to that of Proposition 4.11. For each voter in the Control by Removing Voters instance, we add a voter to V' that gives points to the complement subset of candidates. Formally, denote by $C_v \subseteq C$ the candidates to which v awards points. For each $v \in V$, add v' to V', such that $C_{v'} = C \setminus C_v$. Note that this construction creates a legal Bloc instance since $|C_{v'}| = |C \setminus C_v| = |C| - k = k^*$, and thus

$$(\forall v \in V,\ |C_v| = k) \Rightarrow \left(\forall v' \in V',\ |C_{v'}| = k^*\right).$$

Let $U \subseteq V$; it holds for both voting rules that:

$$s_U[c] = |\{v \in V : c \in C_v\}| - |\{v \in U : c \in C_v\}|$$

and,

$$\begin{aligned}
s^*_U[c] &= |\{v' \in V' : c \in C_{v'}\}| + |\{v \in U : c \in C_v\}| \\
&= |\{v \in V : c \notin C_v\}| + |\{v \in U : c \in C_v\}| \\
&= |V| - |\{v \in V : c \in C_v\}| + |\{v \in U : c \in C_v\}| \\
&= |V| - (|\{v \in V : c \in C_v\}| - |\{v \in U : c \in C_v\}|) \\
&= |V| - s_U[c],
\end{aligned}$$

and thus:

$$\forall c, c' \in C, \forall U \subseteq V, \quad s^*_U[c] \geq s^*_U[c'] \ \Leftrightarrow\ s_U[c] \leq s_U[c'].$$

Cumulative voting: For each original voter, we will add a new voter to V' that distributes his pool in a way that complements the original distribution. Formally, let s^v_c be the score that voter v gives to candidate c; $\forall v \in V,\ \sum_{c\in C} s^v_c = L$. The point distribution of the new voter $v' \in V'$ is $s^{v'}_c = L - s^v_c$, for all $c \in C$. Note that this is a legal Cumulative voting instance, with $L^* = L(m-1)$:

$$\forall v' \in V', \quad \sum_{c\in C} s^{v'}_c = \sum_{c\in C} L - s^v_c = mL - \sum_{c\in C} s^v_c = mL - L = L(m-1) = L^*.$$

Once again we will look at the final score after adding/removing the voters of $U \subseteq V$:

$$s_U[c] = \sum_{v\in V\setminus U} s^v_c = \sum_{v\in V} s^v_c - \sum_{v\in U} s^v_c.$$

Further,

$$\begin{aligned}
s^*_U[c] &= \sum_{v\in V'\uplus U} s^v_c \\
&= \sum_{v'\in V'} s^{v'}_c + \sum_{v\in U} s^v_c \\
&= \sum_{v\in V} (L - s^v_c) + \sum_{v\in U} s^v_c \\
&= L|V| - \sum_{v\in V} s^v_c + \sum_{v\in U} s^v_c \\
&= L|V| - \left(\sum_{v\in V} s^v_c - \sum_{v\in U} s^v_c\right) \\
&= L|V| - s_U[c]
\end{aligned}$$

Therefore, we have:

$$\forall c, c' \in C, \forall U \subseteq V, \quad s^*_U[c] \geq s^*_U[c'] \;\Leftrightarrow\; s_U[c] \leq s_U[c'].$$

□

Our final generic transformation lemma shows that the constructive and destructive settings, i.e., including/excluding a distinguished candidate, are basically equivalent under the three voting rules in question. A subtle point, that we shall deal with after we formulate and prove the lemma, is that the lemma reverses the tie-breaking scheme: adversarial tie-breaking (as we have assumed so far) becomes *friendly* tie-breaking, i.e., ties are broken in favor of candidates with higher utility; and vice versa.

Lemma 4.17. *For every instance of* Control by Adding *(resp.,* Removing) Voters *with adversarial tie-breaking where $W \subseteq C$ can be made to win by adding (resp., removing) r voters, there is an instance of* Control by Adding *(resp.,* Removing) Voters *with friendly tie-breaking with the same candidate set C where $C \backslash W$ can be made to win by adding (resp., removing) r voters. Similarly, friendly tie-breaking is transformed to adversarial tie-breaking. This statement holds in Approval, Bloc, and Cumulative Voting.*

Proof. For clarity, we will prove the lemma for adding voters; the proof for removing voters is practically identical. We will also prove the result for Cumulative voting (with a pool of points of size L), but it is straightforward to extend it to Approval and Bloc, generally by replacing L with 1.

Given an instance $\langle V, V', C, p, k, r\rangle$, let $U \subseteq V'$ be a subset of r voters such that adding these voters induces the set of winners W. Construct an instance of Control by Adding Voters as follows: set the number of winners to be $k^* = |C| - k$, and set the pool of points to be $L^* = L(|C| - 1)$. For each voter $v \in V, V'$ we have a voter v^* who gives $L - s^v_c$ to candidate c, where s^v_c is the score given to c by the voter v. Define the set V^* (resp., V'^*) as the set containing all such voters corresponding to voters in V (resp., V'). Note that

$$\forall v^* \in V^* \uplus V'^*, \quad \sum_{c \in C} s^{v^*}_c = \sum_{c \in C} L - s^v_c = L|C| - \sum_{c \in C} s^v_c = L|C| - L = L(|C| - 1) = L^*.$$

Denote by $\hat{V}$ the final set of voters that results from adding the voters of U, and by $\hat{V}^*$ the corresponding set of voters in the constructed instance (i.e., $v^* \in \hat{V}^* \Leftrightarrow v \in \hat{V}$). Also denote by $s[c], \hat{s}[c]$ the score of each candidate $c \in C$ in the original instance and in the new instance, after adding the voters.

As in previous proofs, we get that the candidates' order has reversed. Formally, when comparing the score of two candidates:

$$\begin{aligned}
\hat{s}[c_1] - \hat{s}[c_2] &= \sum_{v^* \in \hat{V}^*} \hat{s}_{c_1}^{v^*} - \sum_{v^* \in \hat{V}^*} \hat{s}_{c_2}^{v^*} \\
&= \sum_{v \in \hat{V}} (L - s_{c_1}^{v}) - \sum_{v \in \hat{V}} (L - s_{c_2}^{v}) \\
&= |\hat{V}|L - \sum_{v \in \hat{V}} (s_{c_1}^{v}) - |\hat{V}|L + \sum_{v \in \hat{V}} (s_{c_2}^{v}) \\
&= \sum_{v \in \hat{V}} s_{c_2}^{v} - \sum_{v \in \hat{V}} s_{c_1}^{v} \\
&= s[c_2] - s[c_1]
\end{aligned}$$

Therefore, since we assumed the tie-breaking scheme is reversed, the k candidates with highest score in the original instance are the k candidates with the lowest score in the constructed instance, and, moreover, $C \setminus W$ form the $k^* = m - k$ winners of the new instance.[7] □

Lemma 4.17 allows us to derive straightforward reductions between versions of Control where the chairman wants to include/exclude a distinguished candidate: it is possible to include a candidate in a given instance if and only if it is possible to exclude a candidate in the constructed instance, and vice versa. The only point we have to address is the change in the tie-breaking scheme. Fortunately, it is easy to obtain versions of Propositions 4.12, 4.13, and 4.15 under friendly tie-breaking, by slight modifications to the proofs. Hence, the lemma, when applied to these three propositions, yields:

Proposition 4.18. Control by Removing Voters *in Bloc, Approval, and Cumulative voting is $\mathcal{NP}$-complete, even when the chairman simply wants to exclude a distinguished candidate.*

Recall that Lemma 4.10 preserves the tie-breaking scheme. Thus, we can obtain a "friendly" version of Proposition 4.16. Then, Lemma 4.17 applied to the "friendly" Proposition 4.16 gives us the final result of this section. Note that the constructive version (i.e., including a distinguished candidate) of Control by Adding Voters in Approval is already known to be $\mathcal{NP}$-hard even for single winner elections (Hemaspaandra et al., 2007b).

Proposition 4.19. Control by Adding Voters *in Approval (Hemaspaandra et al., 2007b), Bloc, and Cumulative voting is $\mathcal{NP}$-complete, even when the chairman simply wants to include a distinguished candidate.*

Here emerges an issue worth clarifying. As noted above, the constructive version (i.e., including a distinguished candidate) of Control by Adding/Removing Voters in Approval is already known to be $\mathcal{NP}$-hard even for single winner elections (Hemaspaandra et al., 2007b). On the other hand, Hemaspaandra et al. (2007b) show that the *destructive*

7. To be precise, candidates with utilities identical to those of $C \setminus W$ are the winners; the names of the winners may change, but this is irrelevant for our purposes.

version of the same problems (i.e., adding/removing voters while excluding a distinguished candidate) is in $\mathcal{P}$ when $k = 1$. Our surprising results imply that this is no longer the case when the number of winners k is also a parameter.

At first it may appear that it follows from Lemma 4.17 that the constructive and destructive versions of this problem are equivalent, which would seem contradictory. A closer examination reveals that there is no such conflict: if we apply the lemma to the constructive result with $k = 1$, we obtain a reduction to the destructive version, but with $k^* = |C| - 1$.

4.2 Controlling the Set of Candidates

We now turn to the problem of controlling the set of candidates. As noted at the beginning of this section, this problem is ill-defined when it comes to Cumulative voting, so in the following we restrict ourselves to SNTV, Bloc voting, and Approval voting.

Another subtle point is that in the problem of destructive control by removing candidates in single winner elections, it is assumed that the chairman cannot remove the distinguished candidate p. There does not seem to be an elegant way to generalize this assumption to multi-winner elections. Hence, we do not discuss control by removing candidates when the goal is to exclude a distinguished candidate.

Now, we recall that Bartholdi et al. (1992) show that control by adding/removing candidates in SNTV is $\mathcal{NP}$-complete, even in single winner elections (in particular, even when the chairman wants to include a single candidate among the winners). Hemaspaandra et al. (2007b) extended this result to destructive control. Crucially, since these results hold for single winner elections, they also apply to Bloc voting, as Bloc voting and SNTV are identical when $k = 1$. So, in fact, Bartholdi et al. and Hemaspaandra et al. together imply that control by adding candidates in SNTV and Bloc is $\mathcal{NP}$-complete, even when the chairman just wants to include or exclude a distinguished candidate, and control by removing candidates in SNTV and Bloc is $\mathcal{NP}$-complete, even when the chairman just wants to include a distinguished candidate (again, we do not discuss the exclusion of a candidate).

Although Approval voting seems more complicated than SNTV (or even Bloc), surprisingly it is much easier to control by tampering with the set of candidates.

Proposition 4.20. Control by Adding Candidates *in Approval is in $\mathcal{P}$, under any utility function.*

Interestingly, in the single winner setting, Approval is immune to control by adding candidates (Hemaspaandra et al., 2007b), i.e., it is not possible to get a candidate elected by adding other candidates. However, in our multiple winner model it is clearly possible to gain utility by adding candidates.

Proof of Proposition 4.20. We will actually solve the following problem: can the chairman add *exactly* r candidates, in a way that all the added candidates become winners, and the utility is at least t? Solving this problem entails that the chairman can also solve the original problem, as he can simply run the algorithm for every $r' \leq r$.

First note that each candidate in $c \in C, C'$ has a fixed number of points $s[c]$, regardless of the participation of any other candidate. The chairman selects a subset $D \subseteq C'$, $|D| = r$, and the winners are the k candidates in $C \uplus D$ with the highest score. Therefore, it is only effective to add candidates that will actually be winners.

Say indeed that $D \subseteq W$, where W is the set of winners. Regardless of the identity of the candidates in D, the winners from C are exactly the $|C| - r$ candidates with the highest score in C. Since r is fixed and (without loss of generality) $r \le k$, the utility of the winners from C is also fixed, and we only need to optimize D, i.e., select the best r candidates from C' that can be made to be winners.

Lemma 4.21. *Let $C = \{c_1, \ldots, c_m\}$ be sorted by nonincreasing score, then by nondecreasing utility, i.e., $s[c_j] \ge s[c_{j+1}]$ for all $j = 1, \ldots, m-1$, and if $s[c_j] = s[c_{j+1}]$ then $u(c_j) \le u(c_{j+1})$. Let W be the set of winners after the candidates $D \subseteq C'$, $|D| = r$, have been added. Then $D \subseteq W$ if and only if for all $d \in D$, one of the following holds:*

1. $s[d] > s[c_{k-r+1}]$.
2. $s[d] = s[c_{k-r+1}]$ *and* $u(d) < u(c_{k-r+1})$.

Proof. Assume first that $D \subseteq W$. Among the k candidates with highest score, at least r are not from C. Thus, at least r candidates among the highest-scoring k candidates in C are excluded from W; the candidates $\{c_{k-r+1}, \ldots, c_k\}$ are certainly excluded from the set of winners. Since candidates with lower score are excluded first, and equality is broken in favor of candidates with lower utility, we have that either $\forall c \in W$, $s[c_{k-r+1}] < s[c]$, or an equality holds and the utility of c is lower; in particular, this is true for any $d \in D \subseteq W$.

Conversely, suppose that for all $d \in D$, either condition 1 or condition 2 holds. Then exactly k candidates are preferred (by the voting rule and the tie-breaking mechanism) to c_{k-r+1}: $\{c_1, \ldots, c_{k-r}\}$, as well as the candidates in D. Thus these are the k winners. □

Denote *thresh*$(r) = s[c_{k-r+1}]$. By Lemma 4.21, it is sufficient to consider the candidates

$$\{c \in C' : \ s[c] > \textit{thresh}(r) \text{ or } s[c] = \textit{thresh}(r) \ \wedge u(c) < u(c_{k-r+1})\}.$$

Clearly, it is possible to achieve a utility of t if and only if this is accomplished by adding the r candidates with highest utility in this set. □

Proposition 4.22. Control by Removing Candidates *in Approval is in $\mathcal{P}$, under any utility function.*

Proof. This is actually an easier problem than the problem of control by adding candidates, and we give a simpler algorithm: First, sort the candidates by decreasing score, sorting candidates with tied scores by increasing utilities. Next, remove the r candidates with lowest utility out of the first $k + r$ candidates. Finally, check if the total utility of the winner set reaches t.

All candidates ranked below position $k+r$ cannot be elected and are therefore irrelevant from the chairman's perspective. From the additivity of the utility function, it is clear that leaving the k candidates with highest utility maximizes the total utility of the chairman. Thus, our simple algorithm is optimal, and if it fails then we can deduce that control is impossible. □

5. Related Work

Academic interest in the complexity of manipulation in voting was initiated by Bartholdi, Tovey and Trick's (1989) seminal paper. The authors suggested that computational complexity may prevent manipulation in practice, and presented a voting rule, namely Second-Order Copeland, which is hard to manipulate. The paper examined single-winner elections, where the goal of the manipulator is to cast its vote in a way that makes a given candidate win the election.

Bartholdi and Orlin (1991) later proved that the important Single Transferable Vote (STV) rule is $\mathcal{NP}$-hard to manipulate. STV is one of the prominent voting rules in the literature on voting. It proceeds in rounds; in the first round, each voter votes for the candidate it ranks first. In every subsequent round, the candidate with the least number of votes is eliminated, and the votes of voters who voted for that candidate are transfered to the next surviving candidate in their ranking.

Conitzer and Sandholm (2003) examined voting rules that are usually easy to manipulate in single-winner elections, but induced hardness by introducing the notion of a *preround*. In the preround, the candidates are paired; the candidates in each pair compete against each other. The introduction of a preround can make an election $\mathcal{NP}$-hard, $\#\mathcal{P}$-hard, or $\mathcal{PSPACE}$-hard, depending on whether the preround precedes, comes after, or is interleaved with the voting rule, respectively. Elkind and Lipmaa (2005a) generalized this approach using Hybrid voting rules, which are composed of several base voting rules.

Some authors also considered a setting where there is an entire coalition of manipulators. In this setting, the standard formulation of the manipulation problem is as follows: we are given a set of votes that have been cast, and a set of manipulators. In addition, all votes are weighted, e.g., a voter with weight k counts as k voters voting identically. We are asked whether the manipulators can cast their vote in a way that makes a specific candidate win the election.

Conitzer, Sandholm, and Lang (2007) have shown that this problem is $\mathcal{NP}$-hard in a variety of voting rules. Indeed, in this setting the manipulators must coordinate their strategies, on top of taking the weights into account, so manipulation is made much more complicated. In fact, the problem is so complicated that the hardness results hold even when the number of candidates is constant. Hemaspaandra and Hemaspaandra (2007) generalized some of these last results by exactly characterizing the *scoring rules* in which manipulation is $\mathcal{NP}$-hard. Elkind and Lipmaa (2005b) have shown how to use one-way functions to make coalitional manipulation hard.

However, some recent papers have argued that worst-case computational hardness may not be sufficient to prevent manipulation. Indeed, although such hardness implies that the problem has an infinite number of hard instances, it may still be the case that under reasonable real-world distributions over instances, the problem is easy to solve. This theme was discussed by Procaccia and Rosenschein (2007b, 2007a, 2008), by Conitzer and Sandholm (2006), and by Erdélyi et al. (2007).

The complexity of control by a chairman has received somewhat less attention, but nevertheless much is already known. Bartholdi, Tovey and Trick (1992) have studied the complexity of seven different types of control in two voting rules (and a single-winner setting): adding voters/candidates, deleting voters/candidates, partitioning the voters, and

Manipulation

Rule	In/Ex candidate		Boolean utility		General utility
SNTV	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$
Bloc	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$
Approval	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$
Cumulative	$\mathcal{P}$	$\Leftarrow$	$\mathcal{P}$		$\mathcal{NP}$-c

Table 1: The complexity of manipulation in multi-winner elections

two types of candidate partitioning. The authors have reached the conclusion that different voting rules differ significantly in terms of their resistance to control. Hemaspaandra and Hemaspaandra (2007b) extended these results to destructive control, where the chairman wants a candidate *not* to be elected.

Hemaspaandra, Hemaspaandra and Rothe (2007a) have contributed to this investigation by examining twenty different types of control. They showed that in the unique winner setting, there is a voting rule which is resistant to all twenty types. However, it is unclear whether one of the "natural" single-winner voting rules has this property of total resistance to manipulation (though some voting rules come close (Faliszewski, Hemaspaandra, Hemaspaandra, & Rothe, 2007)).

Finally, our structured setting, where voters essentially have preferences over subsets of candidates, is related to recent work on combinatorial voting (see, e.g., Lang (2007) and the references listed there).

Note that this paper subsumes parts of Procaccia, Rosenschein and Zohar (2007) and Meir, Procaccia and Rosenschein (2008).

6. Discussion

The analysis in this paper has focused on the complexity of manipulating and controlling four simple voting schemes which are often considered in the context of multi-winner elections: SNTV, Bloc, Approval, and Cumulative voting. In our formulation of the computational problems, we have assumed the manipulator (or chairman) has some additive utility function over the candidates. We distinguished three major cases: the manipulator wants one candidate to be included in the set of winners, or excluded from the set of winners; the manipulator has a binary utility function; and the manipulator has a general utility function.

At this point we wish to direct the reader's attention to Table 1, which summarizes our results regarding manipulation. A left (respectively right) implication arrow in the table means that the result in the cell is implied by the result in the left (respectively right) cell. For a general utility function, manipulation is hard in Cumulative voting, and easy in the other three. The results about Cumulative voting, however, do not carry over when the manipulator has a binary utility function. Another interesting result, which does not appear above, is that when we restrict ourselves to boolean utility functions, *any* scoring rule can be easily manipulated. This result is formally stated and proven in Appendix A.

Control by Adding/Removing Voters

Rule	In/Ex candidate	Boolean utility	General utility
SNTV	$\mathcal{P}$ [2] $\Leftarrow$	$\mathcal{P}$ $\Leftarrow$	$\mathcal{P}$
Bloc	$\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c
Approval	$\mathcal{NP}$-c*	$\Rightarrow$ $\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c
Cumulative	$\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c

Control by Adding/Removing Candidates

Rule	In/Ex candidate**	Boolean utility	General utility
SNTV	$\mathcal{NP}$-c [1,2]	$\Rightarrow$ $\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c
Bloc	$\mathcal{NP}$-c [1,2]	$\Rightarrow$ $\mathcal{NP}$-c	$\Rightarrow$ $\mathcal{NP}$-c
Approval	$\mathcal{P}$ $\Leftarrow$	$\mathcal{P}$ $\Leftarrow$	$\mathcal{P}$
Cumulative	Irrelevant	Irrelevant	Irrelevant

Table 2: The complexity of control by adding/removing voters/candidates in multi-winner elections. * This result was known for including a candidate (Hemaspaandra et al., 2007b), even in single-winner elections. However, when $k = 1$, destructive control by adding/removing voters in Approval is tractable, while this is not the case when k is a parameter. ** In the context of control by removing candidates, we do not discuss the case of excluding a candidate. [1] Bartholdi et al. (1992). [2] Hemaspaandra et al. (2007b).

It remains an open question which scoring rules (if any) are $\mathcal{NP}$-hard to manipulate under general utility functions.

Table 2 summarizes our results regarding control. Notice that with respect to control, all results are true for the three types of utility functions that appear in the table. Our results imply that control by adding or removing voters is easy in SNTV but hard in the other three rules. Surprisingly, the situation has turned on its head when it comes to control by adding or removing candidates: here it is Approval voting which is easy to control, while other rules are hard. In short, there is no clear hierarchy of resistance to control. We conclude that one has to adopt a voting rule *ad hoc*, depending on whether control by tampering with the set of voters, or with the set of candidates, is the major concern.

Note that some types of control, such as partitioning the set of voters or the set of candidates (Bartholdi et al., 1992), have not been investigated in this paper.

Finally, we wish to connect this work with the ongoing discussion of worst-case versus average-case complexity of manipulation and control in elections. As mentioned in Section 5, a strand of recent research argues that worst-case hardness is not a strong enough guarantee of resistance to strategic behavior, by showing that manipulation problems that are known to be $\mathcal{NP}$-hard are tractable according to different average-case notions (Conitzer & Sandholm, 2006; Procaccia & Rosenschein, 2007a; Zuckerman et al., 2008; Erdélyi et al., 2007). However, that research is still highly inconclusive. Therefore, worst-case complexity of manipulation and control remains an important benchmark for comparing different voting rules, and still inspires a considerable and steadily growing body of work (Conitzer et al., 2007; Hemaspaandra et al., 2007a, 2007b; Faliszewski et al., 2007).

Acknowledgments

The authors wish to thank the anonymous JAIR referees for very helpful comments. This work was partially supported by grant #898/05 from the Israel Science Foundation. Ariel Procaccia is supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

Appendix A. Manipulating Scoring Rules

In Section 3, we have shown that SNTV and Bloc voting, which are both scoring rules, are easy to manipulate under a general utility function. The next proposition establishes that this is true for any scoring rule, under a boolean-valued utility function.

Proposition A.1. *Let P be a scoring rule defined by the parameters $\vec{\alpha} = \langle \alpha_1, \ldots, \alpha_m \rangle$.* Manipulation *in P with any boolean-valued utility function $u : C \to \{0, 1\}$ is in $\mathcal{P}$.*

Proof. Let $\vec{\alpha} = \langle \alpha_1, \ldots, \alpha_m \rangle$ be the parameters of the scoring rule in question. Denote the score of each candidate $c \in C$, before the manipulator has cast his vote, by $s[c]$. Let $\mathcal{J}$ be the manipulator's preference profile, given by:

$$\mathcal{J} = c_{j_1} \succ c_{j_2} \succ \ldots \succ c_{j_m}$$

Suppose some candidate $c \in C$ was ranked in place l by the manipulator, $c = c_{j_l}$. Denote the final score of candidate c, according to the manipulator's profile $\mathcal{J}$, by:

$$s_J[c] = s[c] + \alpha_l$$

Finally, denote the winner set that results from the manipulator's ballot $\mathcal{J}$ by $W_{\mathcal{J}}$.

Lemma A.2. *Given $C' \subseteq C$, $|C| = k$, it is possible to determine in polynomial time if there exists $\mathcal{J}$ s.t. $C' = W_J$.*

Proof. Denote $C' = \{c'_1 \ldots c'_k\}$,

$$C'' = C \setminus C' = \{c''_1, \ldots c''_{m-k}\},$$

where both C', C'' are sorted by nondecreasing score $s[c]$. Let

$$\mathcal{J}^* = c'_1 \succ c'_2 \succ \ldots \succ c'_k \succ c''_1 \succ \ldots \succ c''_{m-k}$$

This preference profile ranks the players in C' first, while giving more points to candidates with lower initial score. Candidates from C'' are ranked next, and the same rule applies. The intuition is that we would like the candidates in C' to have a high-as-possible, more or less balanced, score. Likewise, we would like the candidates in C'' to have a low-as-possible balanced score. This strategy generalizes the algorithm of Bartholdi et al. (1989).

We claim that there exists $\mathcal{J}$ s.t. $C' = W_J$ if and only if $C' = W_{\mathcal{J}^*}$. If $C' = W_{\mathcal{J}^*}$ then obviously there exists $\mathcal{J}$ s.t. $C' = W_J$. Conversely, suppose there exists some $\mathcal{J}^\#$ such that $C' = W_{\mathcal{J}^\#}$. Without loss of generality, this holds (by the adversarial tie breaking assumption)[8] if and only if

$$\forall c' \in C', c'' \in C'',\ s_{\mathcal{J}^\#}[c''] < s_{\mathcal{J}^\#}[c']. \tag{5}$$

We argue that it is possible to obtain $\mathcal{J}^*$ from $\mathcal{J}^\#$ by iteratively transposing pairs of candidates, without changing the winner set. Indeed, we distinguish between three cases:

1. $\exists j_1, j_2 \in \{1, 2, \ldots, k\}$ such that $s[c'_{j_1}] > s[c'_{j_2}]$, but in $\mathcal{J}^\#$ it holds that $c'_{j_1} \succ c'_{j_2}$. Now, transpose the rankings of c'_{j_1} and c'_{j_2} in $\mathcal{J}^\#$, i.e., consider the preference profile which is identical to $\mathcal{J}^\#$ except that the places of c'_{j_1} and c'_{j_2} are switched. Denote by W the new set of winners.

   The score of c'_{j_2} increased, so he is certainly still in W. Moreover, the new final (possibly lower) score of c'_{j_1} is:

   $$s[c'_{j_1}] + \alpha_{j_2} \geq s[c'_{j_2}] + \alpha_{j_2} = s_{\mathcal{J}^\#}[c'_{j_2}]$$

   By (5) we have that:

   $$\forall c'' \in C'',\ s_{\mathcal{J}^\#}[c''] < s_{\mathcal{J}^\#}[c'_{j_2}]$$

   Therefore, $c'_{j_1} \in W$ even after the transposition. We conclude that it still holds that $C' = W$.

2. $\exists j_1, j_2 \in \{1, 2, \ldots, m-k\}$ such that $s[c''_{j_1}] > s[c''_{j_2}]$, but in $\mathcal{J}^\#$ it holds that $c''_{j_1} \succ c''_{j_2}$. A similar argument holds in this case.

3. $\exists c' \in C', c'' \in C''$ such that in $\mathcal{J}^\#$ it holds that $c'' \succ c'$. Clearly the desirable candidate c' can only rank higher if we transpose the two candidates.

Using the three types of transpositions, we can replace a couple of candidates at each step until we obtain $\mathcal{J}^*$ from $\mathcal{J}^\#$. In each such step it remains true that $C' = W$, thus $C' = W_{\mathcal{J}^*}$. □

Lemma A.3. *Given $C' \subseteq C$, $C' \leq k$, it is possible to determine in polynomial time if there exists $\mathcal{J}$ s.t. $C' \subseteq W_{\mathcal{J}}$.*

Proof. Let $C' \subseteq C$, $|C'| = k' < k$. We add to C' the $k - k'$ candidates from C'' with the highest score (according to $s[c]$), and denote this new set of size k by C^*. According to Lemma A.2, we can determine efficiently if there exists $\mathcal{J}$ such that $C^* = W_J$.

We argue that it is enough to check C^*. Indeed, assume that there $\exists \mathcal{J}$ such that $C' \subseteq W_{\mathcal{J}}$. Let $c \in C \setminus W_{\mathcal{J}}$ such that there exists $c' \in W_{\mathcal{J}}$ with $s[c'] < s[c]$. Now, if we transpose, in the ranking $\mathcal{J}$, the candidates c and c', clearly c becomes a winner while c' becomes a loser. Therefore, it is possible to make C^* the set of winners. □

8. Tie breaking works against candidates with utility 1 (which are the ones we ultimately care about), but in favor of candidates in C' with utility 0. However, for ease of exposition, we do not deal with such borderline cases here.

To complete the proof of the proposition, we denote by D the set of candidates whose utility is 1. The total utility is at least t if and only if there is a subset of D of size t that can be included in W. Let C' be the t candidates with the highest score $s[c]$ in D. By similar arguments as before, if this subset cannot win then no other subset of D of size t can. By Lemma A.3 we can efficiently find out whether it is possible to include C' in W. $\square$

References

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

Bartholdi, J., Tovey, C. A., & Trick, M. A. (1989). The computational difficulty of manipulating an election. *Social Choice and Welfare*, *6*, 227–241.

Bartholdi, J., Tovey, C. A., & Trick, M. A. (1992). How hard is it to control an election. *Mathematical and Computer Modelling*, *16*, 27–40.

Brams, S. J., & Fishburn, P. C. (2002). Voting procedures. In Arrow, K. J., Sen, A. K., & Suzumura, K. (Eds.), *Handbook of Social Choice and Welfare*, chap. 4. North-Holland.

Conitzer, V., & Sandholm, T. (2003). Universal voting protocol tweaks to make manipulation hard. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 781–788.

Conitzer, V., & Sandholm, T. (2006). Nonexistence of voting rules that are usually hard to manipulate. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pp. 627–634.

Conitzer, V., Sandholm, T., & Lang, J. (2007). When are elections with few candidates hard to manipulate?. *Journal of the ACM*, *54*, 1–33.

Duggan, J., & Schwartz, T. (2000). Strategic manipulability without resoluteness or shared beliefs: Gibbard-Satterthwaite generalized. *Social Choice and Welfare*, *17*(1), 85–93.

Elkind, E., & Lipmaa, H. (2005a). Hybrid voting protocols and hardness of manipulation. In *16th Annual International Symposium on Algorithms and Computation*, Lecture Notes in Computer Science, pp. 206–215. Springer-Verlag.

Elkind, E., & Lipmaa, H. (2005b). Small coalitions cannot manipulate voting. In *International Conference on Financial Cryptography*, Lecture Notes in Computer Science. Springer-Verlag.

Endriss, U. (2007). Vote manipulation in the presence of multiple sincere ballots. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, pp. 125–134.

Ephrati, E., & Rosenschein, J. S. (1997). A heuristic technique for multiagent planning. *Annals of Mathematics and Artificial Intelligence*, *20*, 13–67.

Erdélyi, G., Hemaspaandra, L. A., Rothe, J., & Spakowski, H. (2007). On approximating optimal weighted lobbying, and frequency of correctness versus average-case polynomial time. Tech. rep., arXiv:cs/0703097v1.

Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2007). Llull and Copeland voting broadly resist bribery and control. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI)*, pp. 724–730.

Ghosh, S., Mundhe, M., Hernandez, K., & Sen, S. (1999). Voting for movies: the anatomy of a recommender system. In *Proceedings of the 3rd Annual Conference on Autonomous Agents (AGENTS)*, pp. 434–435.

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

Haynes, T., Sen, S., Arora, N., & Nadella, R. (1997). An automated meeting scheduling system that utilizes user preferences. In *Proceedings of the 1st International Conference on Autonomous Agents (AGENTS)*, pp. 308–315.

Hemaspaandra, E., & Hemaspaandra, L. (2007). Dichotomy for voting systems. *Journal of Computer and System Sciences*, *73*(1), 73–83.

Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2007a). Hybrid elections broaden complexity-theoretic resistance to control. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1308–1314.

Hemaspaandra, E., Hemaspaandra, L. A., & Rothe, J. (2007b). Anyone but him: The complexity of of precluding an alternative. *Artificial Intelligence*, *171*(5–6), 255–285.

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). *Knapsack Problems*. Springer.

Lang, J. (2007). Vote and aggregation in combinatorial domains with structured preferences. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (AAAI)*, pp. 1366–1371.

Meir, R., Procaccia, A. D., & Rosenschein, J. S. (2008). A broader picture of the complexity of strategic behavior in multi-winner elections. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 991–998.

Procaccia, A. D., Rosenschein, J. S., & Zohar, A. (2007). Multi-winner elections: Complexity of manipulation, control and winner-determination. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1476–1481.

Procaccia, A. D., & Rosenschein, J. S. (2007a). Average-case tractability of manipulation in elections via the fraction of manipulators. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 718–720, Honolulu, Hawaii.

Procaccia, A. D., & Rosenschein, J. S. (2007b). Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research*, *28*, 157–181.

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.

Zuckerman, M., Procaccia, A. D., & Rosenschein, J. S. (2008). Algorithms for the coalitional manipulation problem. In *Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 277–286, San Francisco, California.