The Complexity of Controlling Condorcet, Fallback, and k-Veto Elections by Replacing Candidates or Voters

Marc Neveling, Jörg Rothe, Roman Zorn
Institut für Informatik
Heinrich-Heine-Universität Düsseldorf
Düsseldorf, Germany
{marc.neveling, rothe, roman.zorn}@hhu.de

Abstract

Electoral control models malicious ways of tampering with the outcome of elections via structural changes and has turned out to be one of the central themes in computational social choice. While the standard control types—adding/deleting/partitioning either voters or candidates—has been studied quite comprehensively, much less is known for the control action of replacing voters or candidates. Continuing the work of Loreggia et al. (2014; 2015) and Erdélyi, Reger, and Yang (2018), we study the computational complexity of control by replacing candidates or voters in Condorcet, fallback, and k-veto elections.

1 Introduction

Bartholdi, Tovey, and Trick (1992) were the first to propose control of elections as a malicious way of tampering with their outcome via changing their structure, e.g., by adding or deleting voters or candidates. They introduced the constructive variant where the goal of an election chair is to make a favorite candidate win. Focusing on plurality and Condorcet elections, they studied the complexity of the associated control problems, showing either resistance (NP-hardness) or vulnerability (membership in P). Complementing their work, Hemaspaandra, Hemaspaandra, and Rothe (2007) introduced the destructive variant of control where the chair's goal is to prevent a despised candidate's victory. Pinpointing the complexity of destructive control in plurality and Condorcet, they also studied the constructive and destructive control complexity of approval voting. Since then, plenty of voting rules have been analyzed in terms of their control complexity, as surveyed by Faliszewski and Rothe (2016) and Baumeister and Rothe (2015).

The computational complexity of *replacing* voters or candidates was first studied by Loreggia et al. (2014; 2015) and later on by Erdélyi, Reger, and Yang (2018). Compared with the standard control types (adding/deleting/partitioning voters or candidates), much less is known for the control action of replacing voters or candidates. It can be seen as a combination of adding and deleting them, with the additional constraint that the same number of voters/candidates must be added as have been deleted. Other types of combining control attacks have been investigated by Faliszewski, Hemaspaandra, and Hemaspaandra (2011).

Our contribution is to study the complexity of control by replacing either voters or candidates in Condorcet, fallback, and k-veto elections. The complexity of control under the *standard types* has been studied and completely settled for Condorcet voting, as pointed out above, by Bartholdi, Tovey, and Trick (1992) and Hemaspaandra, Hemaspaandra, and Rothe (2007); for fallback voting by Erdélyi et al. (2015a; 2015b; 2011; 2010); and for veto (i.e., 1-veto) elections by Lin (2012; 2011) (who also settled some cases of standard control in k-veto for $k \geq 2$), Chen et al. (2015), and Maushagen and Rothe (2018; 2017; 2016). Among these rules, fallback voting (a hybrid system due to Brams and Sanver (2009) that combines Bucklin with approval voting) is special in that it is one of the two natural voting rules with a polynomial-time winner problem that are currently known to have the most resistances against standard control attacks, the other one being normalized range voting (Menton 2013).

In the related area of judgment aggregation, control by replacing judges has been introduced by Baumeister et al. (2012) and further studied by Baumeister, Rothe, and Selker (2015).

2 Preliminaries

An election is a pair (C,V) with C being a set of m candidates and V a set of n voters. Voters express their preferences over the candidates by, e.g., linear orders over C, such as $c\ b\ a\ d$ for $C = \{a,b,c,d\}$, where the leftmost candidate is the most preferred one by this voter and preference (strictly) decreases from left to right. A voting rule $\mathcal{R}$ then maps each election (C,V) to a subset $W \subseteq C$ of the candidates, called the $\mathcal{R}$ *winners* (or simply the *winners* if $\mathcal{R}$ is clear from the context) of election (C,V). For candidates $a,b \in C$, denote the number of votes in (C,V) preferring a to b by $N_{(C,V)}(a,b)$. We will study the following voting rules:

- *k-veto*: A candidate gains a point from each vote in which she is ranked higher than in the last k positions (i.e., the candidates in the last k positions are vetoed). The candidate(s) with the most points (i.e., with the fewest vetoes) win(s) the election.
- *Condorcet*: A Condorcet winner is a candidate a who beats all other candidates in pairwise contests, i.e., for

each other candidate b, it holds that $N_{(C,V)}(a,b) > N_{(C,V)}(b,a)$. Note that a Condorcet winner does not always exist, but if there is one, he or she is unique.

- *Fallback*: In a fallback election (C,V), each voter v submits her preferences as a subset of candidates $S_v \subseteq C$ that she approves of and, in addition, a strict linear ordering of those candidates (e.g., if a voter v approves of the candidates $S_v = \{c_1,...,c_k\}$ and orders them lexicographically, her vote would be denoted as $c_1 \cdots c_k \mid C \setminus S_v$). Let $score_{(C,V)}(c) = |\{v \in V \mid c \in S_v\}|$ be the *number of approvals of* c and $score^i_{(C,V)}(c)$ be the *number of level* i *approvals of* c (i.e., the number of voters who approve of c and rank c in their top i positions). The fallback winner(s) will then be determined as follows: (1) A candidate c is a *level* ℓ *winner* if $score^\ell_{(C,V)}(c) > |V|/2$. Letting i be the smallest integer such that there is a level i winner, the candidate(s) with the most level i approvals win(s). (2) If there is no fallback winner on any level, the candidate(s) with the most approvals win(s).

Unlike the original papers on electoral control that in particular investigated the control actions of adding and deleting either candidates or voters (Bartholdi, Tovey, and Trick 1992; Hemaspaandra, Hemaspaandra, and Rothe 2007), we will consider control by *replacing* either candidates or voters, which combines adding and deleting them and was introduced by Loreggia et al. (2014; 2015) and later on also studied by Erdélyi, Reger, and Yang (2018).

For a given voting rule $\mathcal{R}$, define the following problems:

$\mathcal{R}$-CONSTRUCTIVE-CONTROL-BY-REPLACING-CANDIDATES	
Given:	An election $(C \cup D, V)$, where D with $C \cap D = \emptyset$ is a set of spoiler candidates, a distinguished candidate $c \in C$, and an integer $r \in \mathbb{N}$.
Question:	Are there subsets $C' \subseteq C \setminus \{c\}$ and $D' \subseteq D$ of equal size (at most r) such that c is an $\mathcal{R}$ winner of the election $((C \setminus C') \cup D', V)$?

We are given an election $(C \cup D, V)$ in this problem, i.e., all votes in V express preferences over all the candidates in $C \cup D$. But only the candidates from C are taken into account before the control action, and some of those have then been replaced by the same number of candidates from D. In any case, we implicitly assume that missing candidates do not show up in the votes, i.e., all votes from V are restricted to those candidates actually occurring in the election at hand.

$\mathcal{R}$-CONSTRUCTIVE-CONTROL-BY-REPLACING-VOTERS	
Given:	An election $(C, V \cup U)$ with registered voters V, as yet unregistered voters U, a distinguished candidate $c \in C$, and an integer $r \in \mathbb{N}$.
Question:	Are there subsets $V' \subseteq V$ and $U' \subseteq U$ of equal size (at most r) such that c is an $\mathcal{R}$ winner of the election $(C, (V \setminus V') \cup U')$?

In short, we denote the former problem as $\mathcal{R}$-CCRC and the latter as $\mathcal{R}$-CCRV. We will also consider the *destructive* variants of these problems, denoted by $\mathcal{R}$-DCRC and $\mathcal{R}$-DCRV, in which the goal is to prevent the distinguished candidate from being an $\mathcal{R}$ winner. We focus on the so-called *nonunique-winner model* in which we do not care if the distinguished candidate is the *only* winner as long as he or she is *a* winner (respectively, *not even a winner* in the destructive variants). By contrast, in the *unique-winner model* a control action is considered successful only if the distinguished candidate is the *unique* winner (respectively, *not a unique winner*). We note in passing that, with slight modifications, our proofs work for the *unique-winner model* as well.

We assume the reader to be familiar with the basic notions from complexity theory; in particular, with the complexity classes P and NP and the notions of NP-hardness and NP-completeness. For our proofs, we define the following well-known NP-complete problems (Garey and Johnson 1979):

EXACT-COVER-BY-THREE-SETS (X3C)	
Given:	A set $B = \{b_1, b_2, ..., b_{3s}\}$ with $s \geq 1$ and a family $\mathcal{S} = \{S_1, S_2, ..., S_t\}$ of subsets $S_i \subseteq B$ with $\|S_i\| = 3$ for each i, $1 \leq i \leq t$.
Question:	Is there a subfamily $\mathcal{S}' \subseteq \mathcal{S}$ such that every element of B appears in exactly one subset of $\mathcal{S}'$?

HITTING-SET	
Given:	A set $B = \{b_1, b_2, ..., b_{3s}\}$ with $s \geq 1$, a family $\mathcal{S} = \{S_1, S_2, ..., S_t\}$ of subsets $S_i \subseteq B$, and an integer q with $1 \leq q \leq s$.
Question:	Is there a subset $B' \subseteq B$, $\|B'\| \leq q$, such that each $S_i \in \mathcal{S}$ is *hit* by B' (i.e., $S_i \cap B' \neq \emptyset$ for all $S_i \in \mathcal{S}$)?

We call a voting rule *immune* to a type of control if it is never possible for the chair to reach her goal by this control action; otherwise, the voting rule is said to be *susceptible* to this control type. A susceptible voting rule is said to be *vulnerable* to this control type if the associated control problem is in P, and it is said to be *resistant* to it if the associated control problem is NP-hard. Note that all considered control problems are in NP, so resistance implies NP-completeness.

3 Overview of Results

Table 1 gives an overview of our results.

Problem	Condorcet	Fallback	k-Veto
CCRV	R	R	V ($k \leq 2$)† / R ($k \geq 3$)†
DCRV	V	V	V ($k \geq 1$)†
CCRC	V	R	R ($k \geq 1$)‡
DCRC	V	R	R ($k \geq 1$)‡

Table 1: Overview of complexity results. "R" stands for *resistant* and "V" for *vulnerable*. Results marked by "†" are due to Erdélyi, Reger, and Yang (2018) and "‡" means that the case $k = 1$ is due to Loreggia et al. (2015).

4 Condorcet Voting

We will start with Condorcet and show that it is vulnerable to three types of control, yet resistant to the forth one.

Theorem 1 *Condorcet is resistent to constructive control by replacing voters.*

Proof. We prove NP-hardness by reducing X3C to Condorcet-CCRV. A similar reduction was used by Bartholdi, Tovey, and Trick (1992) to prove that Condorcet-CCAV (where CCAV stands for "constructive control by adding voters") is NP-hard.

Let $(B,\mathcal{S})$ be an X3C instance with $B=\{b_1,\ldots,b_{3s}\}$, $s\geq 2$ (which may be assumed, as X3C is trivially solvable when $s=1$), and $\mathcal{S}=\{S_1,\ldots,S_t\}$, $t\geq 1$. The set of candidates is $C=B\cup\{c\}$ with c being the distinguished candidate. The list V of votes is constructed as follows:

- There are $2s-3$ registered votes of the form $b_1\ \cdots\ b_{3s}\ c$ in V and
- for each j, $1\leq j\leq t$, there is one unregistered vote of the form $S_j\ c\ B\setminus S_j$ in U.

The ordering of candidates in S_j and $B\setminus S_j$ does not matter in any of those votes. Finally, set $r=s$.

Analyzing the election (C,V), b_1 is the Condorcet winner; in particular, c loses against every $b_i\in B$ with a deficit of $2s-3$ votes, i.e.,

$$N_{(C,V)}(b_i,c)-N_{(C,V)}(c,b_i)=2s-3.$$

We will now show that $(B,\mathcal{S})$ is a yes-instance of X3C if and only if c can be made the Condorcet winner of the election by replacing s votes from V with votes from U.

From left to right, assume there is an exact cover $\mathcal{S}'\subseteq\mathcal{S}$ of B. We remove s votes of the form $b_1\ \cdots\ b_{3s}\ c$ from the election and replace them by the votes of the form $S_j\ c\ B\setminus S_j$ for all $S_j\in\mathcal{S}'$. Let (C,V') be the resulting election. Since $\mathcal{S}'$ is an exact cover of B, for each $b_i\in B$,

$$N_{(C,V')}(b_i,c)-N_{(C,V')}(c,b_i)=(2s-3-s+1)-(s-1)=-1<0.$$

Thus c now defeats each $b_i\in B$ in pairwise comparison and, therefore, has been made the Condorcet winner of (C,V').

From left to right, assume that c can be made a Condorcet winner of the election by replacing at most s votes. Recall that c has a deficit of

$$N_{(C,V)}(b_i,c)-N_{(C,V)}(c,b_i)=2s-3$$

to every $b_i\in B$ in the original election. Thus *exactly* s votes need to be removed from the election, for otherwise c's deficit of at least $s-2$ to every other candidate cannot be caught up on, since at least one other candidate is in front of c in every unregistered vote. With s removed votes, c's deficit to every other candidate is now decreased to $s-3$. However, none of the s votes from U replacing the removed votes can rank some $b_i\in B$ in front of c more than once, as otherwise we would have

$$N_{(C,V')}(b_i,c)\geq s-1\quad\text{ and }\quad N_{(C,V')}(c,b_i)\leq s-2$$

for at least one $b_i\in B$ in the resulting election (C,V'), and c would not win. Let $\mathcal{S}'\subseteq\mathcal{S}$ be the set such that each $S_j\in\mathcal{S}'$ corresponds to the vote $S_j\ c\ B\setminus S_j$ from U that is added to the election to replace a removed vote. Every unregistered voter ranks three candidates of B in front of c. By the pigeonhole principle, in order for the s new votes to rank each of the $3s$ candidates of B in front of c only once, $\mathcal{S}'$ needs to be an exact cover of B. ❑

Theorem 2 *Condorcet is vulnerable to destructive control by replacing voters.*

Proof. To prove membership in P, we will provide an algorithm that solves the problem in polynomial time and outputs, if possible, which of the registered voters must be replaced by which unregistered voters for c to not win.

The input to our algorithm is an election $(C,V\cup U)$, the distinguished candidate $c\in C$, and an integer r. The algorithm will output either a pair (D,A) with $D\in V$, $A\in U$ and $|D|=|A|\leq r$ (i.e., in D are voters that must be removed and in A are voters that must be added to the election instead for c to not win), or that control is impossible. First, the algorithm checks whether c is already not winning the election (C,V) and outputs $(\emptyset,\emptyset)$ if this is the case, and we are done. Otherwise, c currently wins, and the algorithm iterates over all candidates $d\in C\setminus\{c\}$ and first checks whether $N_{(C,V)}(c,d)-N_{(C,V)}(d,c)+1\leq 2r$ (if this is not the case, d loses to c in any case and we can skip this candidate.) Let $D\subseteq V$ contain at most r votes from V preferring c to d and let $A\subseteq U$ contain at most r votes from U preferring d to c. If one of them is smaller than the other, remove votes from the larger one until they are equal in size.

Then we check whether

$$N_E(C,(V\cup A)\setminus D)(c,d)\leq N_E(d,c)$$

in the election $E=(C,(V\cup A)\setminus D))$. If this is the case, c does not beat d in direct comparison, so c cannot win the election. The algorithm then outputs (D,A).

Otherwise, d cannot beat c and the algorithm proceeds to the next candidate. If, after all iterations, no candidate was found that beats or ties c, the algorithm outputs "control impossible."

Obviously, this algorithm runs in polynomial-time and solves the problem. ❑

Bartholdi, Tovey, and Trick (1992) observed that, due to the Weak Axiom of Revealed Preference, Condorcet voting is immune to constructive control by adding candidates, and Hemaspaandra, Hemaspaandra, and Rothe (2007) made the same observation regarding destructive control by deleting candidates. For control by *replacing* candidates, however, Condorcet is susceptible both in the constructive and in the destructive case.

In the constructive case, for instance, if $C=\{b,c\}$ and there is one spoiler candidate in $D=\{d\}$ and only one vote $b\ c\ d$ over $C\cup D$, we can turn c (who does not win according to $b\ c$) into a Condorcet winner by replacing b with d (so we now have $c\ d$).

For susceptibility in the destructive case, just consider $C'=\{c,d\}$ and $D'=\{b\}$, and replace d with b, all else being equal.

Moreover, since in Condorcet elections the direct comparison between two candidates cannot be influenced by deleting or adding other candidates to the election, Condorcet-CCRC and Condorcet-DCRC are both easy to solve.

Theorem 3 *Condorcet is vulnerable to constructive control by replacing candidates.*

Proof. To prove membership in P, we will provide an algorithm that solves the problem in polynomial time and outputs, if possible, which of the original candidates must be replaced by which spoiler candidates for c to win.

The input to our algorithm is an election $(C \cup C', V)$, the distinguished candidate $c \in C$, and an integer r. The algorithm will output either a pair (D, A) with $D \subseteq C \setminus \{c\}$, $A \subseteq C'$ and $|D| = |A| \leq r$ (i.e., in D are candidates that must be removed and in A are candidates that must be added to the election for c to win), or that control is impossible.

First, we check whether c already wins the election (C, V) and output $(\emptyset, \emptyset)$ if this is the case, and we are done.

Otherwise, let $D \subseteq C \setminus \{c\}$ be the set of candidates from $C \setminus \{c\}$ that beat or tie c in direct comparison and let $A \subseteq C'$ be a set of at most $|D|$ candidates from C' that c beats in direct comparison.

If $|D| \leq r$ and $|D| = |A|$, we output (D, A), and otherwise we output "control impossible."

Obviously, the algorithm solves the problem and runs in polynomial time. ❑

Theorem 4 *Condorcet is vulnerable to destructive control by replacing candidates.*

Proof. An algorithm that solves the problem works as follows: Given an election $(C \cup C', V)$, a distinguished candidate $c \in C$, and an integer r, it checkes whether c is not winning the election (C, V) and outputs $(\emptyset, \emptyset)$ if this is the case.

Otherwise, it checks whether there is a candidate $d \in C'$ who beats or ties c in direct comparison, whether there is another candidate $b \in C$ with $b \neq c$ and whether $r \geq 1$. If these conditions are satisfied, it outputs $(\{b\}, \{d\})$, and otherwise "control impossible".

This algorithm outputs either a successful pair (D, A) with $D \subseteq C \setminus \{c\}$, $A \in C'$, and $|D| = |A| \leq r$ if c can be prevented from winning by replacing at most r candidates, or else "control impossible."

Obviously, the algorithm is correct and runs in polynomial time. ❑

5 Fallback Voting

We will now consider fallback voting and show that it is vulnerable to one type of control and resistant to the others.

Theorem 5 *Fallback is resistent to constructive control by replacing voters.*

Proof. To prove NP-hardness, we will modify the reduction from X3C that Erdélyi and Rothe (2010) (and Erdélyi et al. (2015a)) used to show NP-hardness of fallback-CCAV.

Let $(B, \mathcal{S})$ be an X3C instance with $B = \{b_1, \ldots, b_{3s}\}$, $s \geq 2$, and $\mathcal{S} = \{S_1, \ldots, S_t\}$, $t \geq 1$. The set of candidates is $C = B \cup D \cup \{c\}$ with c being the distinguished candidate and $D = \{d_1, \ldots, d_{t(3s-4)}\}$ a set of $t(3s-4)$ dummy candidates. In V (corresponding to the registered voters), there are the $3s-1$ votes:

- $2s-1$ votes of the form $B \mid D \cup \{c\}$ and
- for each i, $1 \leq i \leq s$, one vote $d_i \mid B \cup (D \setminus \{d_i\}) \cup \{c\}$.

In U (corresponding to the unregistered voters), there are the following t votes:

- For each j, $1 \leq j \leq t$, let

$$D_j = \{d_{(j-1)(3s-4)+1}, \ldots, d_{j(3s-4)}\}$$

and include in U the vote

$$D_j \; S_j \; c \mid (B \setminus S_j) \cup (D \setminus D_j).$$

Finally, set $r = s$.

Having no approvals in (C, V), c does not win. We will show that $(B, \mathcal{S})$ is a yes-instance of X3C if and only if c can be made a fallback winner of the constructed election by replacing at most s votes from V with as many votes from U.

From left to right, suppose there is an exact cover $\mathcal{S}' \subseteq \mathcal{S}$ of B. Remove s votes $B \mid D \cup \{c\}$ from the election and add, for each $S_j \in \mathcal{S}'$, the vote $D_j \; S_j \; c \mid (B \setminus S_j) \cup (D \setminus D_j)$ instead. Let $(C, \widehat{V})$ be the resulting election. It follows that

- $score_{(C,\widehat{V})}(d_i) \leq 2$ for every $d_i \in D$,
- $score_{(C,\widehat{V})}(b_i) = s$ for every $b_i \in B$ ($s-1$ approvals from the non-removed registered voters and one approval from the added voters since $\mathcal{S}'$ is an exact cover of B), and
- $score_{(C,\widehat{V})}(c) = s$.

Thus no candidate has a majority on any level and c is one of the winners since she ties all candidates of B for the most approvals overall.

From right to left, suppose c can be made a fallback winner of the election by replacing at most s votes from V with as many votes from U. Since c has no approvals in (C, V) and we can only add at most s approvals for c, the only chance for c to win is to have the most approvals in the last stage of the election. Regardless of which votes we remove or add to the election, every dummy candidate can have at most two approvals, which will at least be tied by c if we add $s \geq 2$ unregistered votes to the election. We need to remove s votes $B \mid D \cup \{c\}$ from the election; otherwise, some $b_i \in B$ would have at least s approvals, whereas c could gain no more than $s-1$ approvals from adding unregistered votes. Each $b_i \in B$ receives $s-1$ approvals from the remaining registered votes of the original election and c reveices s approvals from the added votes. Additionally, every added voter approves of three candidates from B. Hence, in order for c to at least tie every candidate from B, each $b_i \in B$ can only be approved by at most one of the added votes. Since there are s added votes, there must be an exact cover of B. ❑

Theorem 6 *Fallback is vulnerable to destructive control by replacing voters.*

Proof. We provide a polynomial-time algorithm that solves the problem and computes which voters need to be removed and which need to be added to make the distinguished candidate a fallback winner. The algorithm is inspired by an algorithm designed by Erdélyi and Rothe (2010) (see also Erdélyi et al. (2015a)) to prove membership of fallback-DCAV in P.

For an election (C,V), let $maj(V) = \lfloor |V|/2 \rfloor + 1$ and let

$$def^i_{(C,V)}(d) = maj(V) - score^i_{(C,V)}(d)$$

be the deficit of candidate $d \in C$ to a strict majority in (C,V) on level i, $1 \leq i \leq |C|$. Note that the number of voters is always the same, namely $|V|$, and so we will use $maj(V)$ even after we have replaced some voters.

The input of the algorithm is an election $(C, V \cup U)$, a distinguished candidate $c \in C$, and an integer r. The algorithm will output either a pair (D,A) with $D \subseteq V$, $A \subseteq U$, and $|D| = |A| \leq r$ (i.e., in D are votes that must be removed and in A are votes that must be added to the election for c to not win), or that control is impossible.

The algorithm runs through $n = \max_{v \in V \cup U} |S_v|$ stages which we call the *majority stages* and one final stage which we call the *approval stage*. In the majority stages the algorithm checks whether c can be beaten in the first n levels of the fallback election by replacing at most r voters, and in the approval stage it checks whether c can be dethroned in the last stage of the fallback election by this control action.

The algorithm works as follows: If c is already not winning in (C,V), we output $(\emptyset,\emptyset)$ and are done.

Majority Stage 1: For every candidate $d \in C \setminus \{c\}$, we check whether d can beat c on the first level by replacing at most r voters. For this the following must hold:

$$def^1_{(C,V)}(d) \leq r; \quad (1)$$

$$score^1_{(C,V)}(d) > score^1_{(C,V)}(c) - 2r. \quad (2)$$

If at least one of (1) and (2) does not hold, d can never have a strict majority on level one or cannot beat c on level one, no matter which r votes we replace, and we can skip d and proceed to the next candidate (or to the next stage if all candidates failed to beat c in this stage). Otherwise, we determine the largest $U_d \subseteq U$ such that $|U_d| \leq r$ and all votes of U_d approve of d on the first level.

Furthermore, we determine the largest $V_d \subseteq V$ such that $|V_d| \leq r$ and all votes of V_d approve of c on the first level. If $|V_d| \neq |U_d|$, we fill up the smaller vote list with votes as follows until they are equal in size.

If V_d is the smaller list, we first choose votes of $V \setminus V_d$ who approve of other candidates than d and add them to V_d; and if U_d is the smaller list, we first choose votes of $U \setminus U_d$ who approve of other candidates than c and add them to U_d. Only when we run out of votes to fill up the smaller list before both lists are equal in size, we will remove voters from the larger list until they are equal in size.

Then we check whether the following conditions hold:

$$score^1_{(C,(V\setminus V_d)\cup U_d)}(d) \geq maj(V); \quad (3)$$

$$score^1_{(C,(V\setminus V_d)\cup U_d)}(d) > score^1_{(C,(V\setminus V_d)\cup U_d)}(c). \quad (4)$$

If at least one of (3) and (4) does not hold, we skip d and proceed to the next candidate (or, if none is left, to the next stage).

Otherwise, we output (V_d, U_d).

Majority Stage i, $2 < i \leq n$: This stage is reached if we could not control the election in stages 1 through $i-1$. Now, for every candidate $d \in C \setminus \{c\}$, we check whether d can beat c on level i of the fallback election. First, we check if the following two equations hold:

$$def^i_{(C,V)}(d) \leq r; \quad (5)$$

$$score^i_{(C,V)}(d) > score^i_{(C,V)}(c) - 2r. \quad (6)$$

If at least one of (5) and (6) does not hold, d can never have a strict majority on level i or cannot beat c on this level, no matter which r votes we replace, and we skip d and proceed to the next candidate (or the next stage if all candidates failed to beat c in this stage).

Otherwise, we determine the largest $U_d \subseteq U$ such that $|U_d| \leq r$ and all votes of U_d approve of d and disapprove of c on the first i levels.

Furthermore, we determine the largest $V_d \subseteq V$ such that $|V_d| \leq r$ and all votes of V_d approve of c and disapprove of d on the first i levels.

If $|V_d| < |U_d|$, we fill up V_d with votes of $V \setminus V_d$ who approve of neither c nor d until we either have $|V_d| = |U_d|$ or run out of those votes, and in the latter case we now keep adding to V_d those votes of $V \setminus V_d$ who approve of both c and d while prioritizing those votes that approve of c on levels up to $i-1$ over votes that approve of c on level i. Only if this is still not enough to make these two vote lists equal in size, we remove votes from U_d until they are equally large.

If $|V_d| > |U_d|$, we fill up U_d with votes of $U \setminus U_d$ that approve of both c and d on the first i levels while prioritizing those votes that approve of c on level i over votes that approve of c on levels up to $i-1$, and if this is not enough to make these two vote lists equal in size, we add those votes from $U \setminus U_d$ to U_d that disapprove of both c and d. Again, only if this is still not enough to make them both equal in size, we will remove votes from V_d while prioritizing votes that approve of c on level i.

Now, knowing that the resulting lists V_d and U_d are equal in size, we check the following condition:

$$score^i_{(C,(V\setminus V_d)\cup U_d)}(d) > score^i_{(C,(V\setminus V_d)\cup U_d)}(c). \quad (7)$$

If (7) does not hold or d does not have a strict majority on the first i levels in $(C,(V \setminus V_d) \cup U_d)$, d cannot beat c and win on level i, and we skip d and proceed to the next candidate or the next stage.

Otherwise, we check the following condition:

$$score^{i-1}_{(C,(V\setminus V_d)\cup U_d)}(c) \geq maj(V). \quad (8)$$

If (8) does not hold, we output (V_d, U_d), as d wins on the ith level and so prevents c from winning. If (8) does hold, then c wins on an earlier level and we failed to control the election. We will try to fix this, if at all possible, in two steps.

Firstly, if there are votes in U_d that approve of c on levels up to $i-1$ and of d on the first i levels (this would mean that all votes in V_d approve of c and disapprove of d on the first i levels), then we remove, by taking turns, one of them from U_d and one from V_d that approve of c on level i as long as possible and as long as

$$score^i_{(C,(V\setminus V_d)\cup U_d)}(d) \geq maj(V)$$

and (7) still hold.

Secondly, we find the largest vote lists $U_{cd} \subseteq (U \setminus U_d)$ and $V_{cd} \subseteq (V \setminus V_d)$ such that:

- $|V_d \cup V_{cd}| \leq r$,
- $|V_{cd}| = |U_{cd}|$,
- all votes in V_{cd} approve of c on the first $i-1$ levels and of d on the first i levels,
- all votes in U_{cd} approve of c on level i and of d on the first i levels, or disapprove of both c and d on the first i levels, and
- $score^{i}_{(C,(V\setminus(V_d\cup V_{cd}))\cup U_d\cup U_{cd})}(d) \geq maj(V)$.

Then we check the following condition:

$$score^{i-1}_{(C,(V\setminus(V_d\cup V_{cd}))\cup U_d\cup U_{cd})}(c) \quad \geq \quad maj(V). \tag{9}$$

If (9) holds, c cannot be prevented from reaching a strict majority in the first i levels without d not reaching a strict majority as well.

Otherwise, d still has a strict majority on level i and c cannot beat d with a strict majority on earlier levels, so we output $(V_d \cup V_{cd}, U_d \cup U_{cd})$ as a successful pair.

Approval Stage: This stage will only be reached if it was not possible to find a control action in the *majority stages* 1 through n. We first check whether the following holds:

$$score_{(C,V)}(c) - r < maj(V). \tag{10}$$

If (10) does not hold, we output "control impossible" since, after replacing at most r suitable votes, (1) we could not find a candidate that beats c in the majority stages and reaches a strict majority and (2) c cannot be prevented from reaching a strict majority in overall approvals; so c must win, no matter which at most r votes are replaced.

Otherwise, we iterate over all candidates $d \in C \setminus \{c\}$ and check whether $score_{(C,V)}(c) - 2r > score_{(C,V)}(d)$. If this is not the case, we skip d and proceed to the next candidate or, if none is left, we output "control impossible" since then d cannot catch up on her deficit to c.

Otherwise, we will try to make d overtake c in overall approvals while decreasing c's overall approvals as much as possible in order to prevent c from reaching a strict majority. We again determine the largest $U_d \subseteq U$ such that $|U_d| \leq r$ and all votes of U_d approve of d and disapprove of c. Furthermore, we again determine the largest $V_d \subseteq V$ such that $|V_d| \leq r$ and all votes of V_d approve of c and disapprove of d.

If $|V_d| < |U_d|$, we fill up V_d with votes of $V \setminus V_d$ who approve of both c and d until we either have $|V_d| = |U_d|$ or run out of those votes, and in the latter case we now keep adding to V_d those votes of $V \setminus V_d$ who approve of neither c nor d. Only if this is still not enough to make the two lists equal in size, we remove votes from U_d until they are equally large.

If $|V_d| > |U_d|$, we fill up U_d with votes of $U \setminus U_d$ that disapprove of both c and d until we either have $|V_d| = |U_d|$ or run out of those votes, and in the latter case we now keep adding to U_d those votes of $U \setminus U_d$ that approve of both c and d. Again, only if this is still not enough to make both vote lists equal in size, we remove votes from V_d until they are equally large. Afterwards, if there are votes in $V \setminus V_d$ that approve of both c and d and votes in $U \setminus U_d$ that disapprove of both c and d, we add as many as possible of them to V_d and U_d, respectively, always ensuring that $|V_d| = |U_d|$ still holds.

Then we check the following conditions:

$$score_{(C,(V\setminus V_d)\cup U_d)}(d) \quad > \quad score_{(C,(V\setminus V_d)\cup U_d)}(c), \tag{11}$$

$$score_{(C,(V\setminus V_d)\cup U_d)}(c) \quad < \quad maj(V). \tag{12}$$

If (11) and (12) are true, output (V_d, U_d) since we have prevented c from reaching a strict majority and found a candidate d that beats c. Otherwise, we skip to the next candidate or, if none is left, output "control impossible."

Correctness of the algorithm follows from the explanations given during its description: The algorithm takes the safest way possible to guarantee that a yes-instance is verified. Clearly, the algorithm runs in polynomial time. ❑

Turning to control by replacing candidates, fallback is resistant in both the constructive and the destructive case.

Theorem 7 *Fallback is resistent to constructive and destructive control by replacing candidates.*

Proof. Erdélyi and Rothe (2010) (see also the subsequent journal version by Erdélyi et al. (2015a)) showed that fallback is resistant to constructive and destructive control by deleting candidates. In the former problem (denoted by fallback-CCDC), we are given a fallback election (C,V), a distinguished candidate $c \in C$, and an integer r, and we ask whether c can be made a fallback winner by deleting at most r votes. In the destructive variant (denoted by fallback-DCDC), for the same input we ask whether we can prevent c from winning by deleting at most r votes. To prove the theorem, we will reduce

- fallback-CCDC to fallback-CCRC and
- fallback-DCDC to fallback-DCRC, respectively.

Let $((C,V),c,r)$ be an instance of fallback-CCDC (or fallback-DCDC). We construct from (C,V) a fallback election $(C \cup D, V')$ with (dummy) spoiler candidates $D = \{d_1,\ldots,d_r\}$, $D \cap C = \emptyset$, where we extend the votes of V to the set of candidates $C \cup D$ by letting all voters disapprove of all candidates in D, thus obtaining V'. Our distinguished candidate remains c, and r remains the limit on the number of candidates that may be replaced.

Since all candidates from D are irrelevant to the election and can be added to the election without changing the winner(s), it is clear that c can be made a fallback winner of (C,V) by deleting up to r candidates from C if and only if c can be made a fallback winner of $(C \cup D, V')$ by deleting up to r candidates from C and adding the same number of dummy spoiler candidates from D. This gives the desired reduction in both the constructive and the destructive case. ❑

6 *k*-Veto

Erdélyi, Reger, and Yang (2018) solved the two cases of control by replacing voters for k-veto (recall Table 1 in Section 3), while Loreggia et al. (2015) solved the two cases of

c	d	$c' \in C'$	$y \in Y$	$x \in X$
$M(s+1)+sq+t$	$M+q$	M	0	$M(s+1)+2s+q+1$

control by replacing candidates for veto only (i.e., for k-veto with $k = 1$). We solve these cases for k-veto with $k \geq 2$.

Theorem 8 *For $k \geq 2$, k-veto is resistent to constructive control by replacing candidates.*

Proof. To prove NP-hardness of k-veto-CCRC for $k \geq 2$, we will modify the reduction provided by Lin (2011) to prove that k-veto-CCAC and k-veto-CCDC are NP-hard. Since his reduction was designed so as to prove both cases at once but we only need the "adding candidates" part, we will simplify the reduction.

Let $(B,\mathcal{S},q)$ be an instance of HITTING-SET with $B = \{b_1,\ldots,b_s\}$, $s \geq 1$, $\mathcal{S} = \{S_1,\ldots,S_t\}$, $t \geq 1$, and integer q, $1 \leq q < s$ (without loss of generality, we may assume that $q < s$ since $(B,\mathcal{S},q)$ is trivially a yes-instance if $q \geq s$).

We construct an instance $((C \cup B, V), c, q)$ of k-veto-CCRC with candidates $C = \{c,d\} \cup C' \cup X \cup Y$, where

$$\begin{aligned} C' &= \{c'_1,\ldots,c'_{k-1}\}, \\ X &= \{x_1,\ldots,x_{k-1}\}, \text{ and} \\ Y &= \{y_1,\ldots,y_q\}, \end{aligned}$$

and spoiler candidates B. Let V contain the following votes:

- $(t+2s)(s-q+1)$ votes $Y \cdots c\ c'_1 \cdots c'_{k-1}$;
- $(t+2s)(s-q+1)-s+q$ votes $Y \cdots d\ x_1 \cdots x_{k-1}$;
- for each i, $1 \leq i \leq t$, one vote $Y \cdots c\ x_1 \cdots x_{k-1}\ S_i$;
- for each i, $1 \leq i \leq s$, one vote $Y \cdots d\ x_1 \cdots x_{k-1}\ b_i$; and
- for each i, $1 \leq i \leq s$, $(t+2s)(s-q+1)+q$ votes $Y \cdots c\ B \setminus \{b_i\}\ x_1 \cdots x_{k-1}\ b_i$.

Let $M = (t+2s)(s-q+1)$. Without the spoiler candidates, vetoes are assigned to the other candidates as follows:

We show that $(B,\mathcal{S},q)$ is a yes-instance of HITTING-SET if and only if c can be made a k-veto winner of the election by replacing q candidates from C with candidates from B.

From left to right, assume there is a hitting set $B' \subseteq B$ of $\mathcal{S}$ of size q (since $q < s$, if B' is a hitting set of size less than q, we fill B' up by adding arbitrary candidates from $B \setminus B'$ to B' until $|B'| = q$). We then replace the candidates from Y with the candidates from B'. Since c, d, and candidates from C' have $(t+2s)(s-q+1)$ vetoes and candidates from X and B' have at least $(t+2s)(s-q+1)+q$ vetoes, c is a k-veto winner.

From right to left, assume c can be made a k-veto winner of the election by replacing q candidates. Since the q candidates from Y have zero vetoes but c has at least one veto, we need to remove each candidate of Y (and no other candidate), and in turn we need to add q candidates from B. Note that c cannot have more than $(t+2s)(s-q+1)$ vetoes, for otherwise c would lose to the candidates from C'. Let $B' \subseteq B$ be the set of q candidates from B that are added to the election. Since $|B'| = q > 0$, c will lose all $s((t+2s)(s-q+1)+q)$ vetoes from the last group of voters. Furthermore, in order to tie the candidates in C', c cannot gain any vetoes from the third group of voters. Thus the q added candidates from B need to be a hitting set of $\mathcal{S}$. Also note that with the q added candidates from B, c also ties d (who lost q vetoes from the fourth group of voters) and beats the candidates from X and the added candidates from B. ❑

Theorem 9 *For $k \geq 2$, k-veto is resistent to destructive control by replacing candidates.*

Proof. As in the proof of Theorem 8, we will prove NP-hardness of k-veto-DCRC, $k \geq 2$, by providing a reduction from HITTING-SET to k-veto-DCRC that is a simplified and slightly modified variant of a reduction used by Lin (2011) to show that k-veto-DCAC and k-veto-DCDC are NP-hard.

Let $(B,\mathcal{S},q)$ be an instance of HITTING-SET with $B = \{b_1,\ldots,b_s\}$, $s \geq 1$, $\mathcal{S} = \{S_1,\ldots,S_t\}$, $t \geq 1$, and integer q, $1 \leq q \leq s$. We construct an instance $((C \cup B, V), c, q)$ of k-veto-DCRC with candidates $C = \{c,c'\} \cup X \cup Y$, where $X = \{x_1,\ldots,x_{k-1}\}$ and $Y = \{y_1,\ldots,y_q\}$, and spoiler candidates B. Let V contain the following votes:

- $2(s-q)+2t(q+1)+4$ votes $\cdots\ c\ Y\ x_1 \cdots x_{k-1}\ c'$;
- $2t(q+1)+5$ votes $\cdots\ c'\ x_1 \cdots x_{k-1}\ c$;
- for each i, $1 \leq i \leq t$, $2(q+1)$ votes $\cdots\ c'\ x_1 \cdots x_{k-1}\ S_i$;
- for each i, $1 \leq i \leq s$, two votes $\cdots\ c\ Y\ x_1 \cdots x_{k-1}\ b_i$;
- for each i, $1 \leq i \leq q$, $2(s-q)+2t(q+1)+6$ votes $c\ c' \cdots y_i\ x_1 \cdots x_{k-1}$; and
- for each i, $1 \leq i \leq s$, $2(s-q)+2t(q+1)+6$ votes $c\ c' \cdots b_i\ x_1 \cdots x_{k-1}$.

In (C,V), c wins the election with $2t(q+1)+5$ vetoes while c' has $2(s-q)+4t(q+1)+4$ vetoes and every other candidate has at least $2(s-q)+2t(q+1)+6$ vetoes.

To complete the proof of Theorem 9, we will now show that $(B,\mathcal{S},q)$ is a yes-instance of HITTING-SET if and only if c can be prevented from being a k-veto winner of the election by replacing q candidates from C with candidates from B.

From left to right, assume there is a hitting set $B' \subseteq B$ of $\mathcal{S}$ of size q (since $q < s$, if B' is a hitting set of size less than q, we fill B' up by adding arbitrary candidates from $B \setminus B'$ to B' until $|B'| = q$). Replacing the candidates from Y with the candidates from B', c gains $2(s-q)$ vetoes and now has $2(s-q)+2t(q+1)+5$ vetoes and c' loses $2t(q+1)$ vetoes and now has $2(s-q)+2t(q+1)+4$ vetoes, so c does no longer win the election.

From right to left, assume c can be prevented from being a k-veto winner of the election by replacing at most q candidates. We first argue why we must remove all q candidates from Y. Firstly, from removing c' from the election, c's strongest rival, c does not gain any vetoes and then there won't be any candidate in the election that can beat c. Secondly, removing any candidate in X from the election will lead to c' gaining vetoes (which c' cannot afford) while c can in the best case gain the same number of vetoes as c would gain by replacing candidates from Y. Thus removing candidates from Y is the best choice. All q candidates from Y need to be removed, for otherwise c does not gain any vetoes. Then q candidates from B need to be added to the election. Note that c will always gain $2(s-q)$ vetoes from those

replacements, which will bring c to $2(s-q)+2t(q+1)+5$ vetoes, so every candidate other than c' cannot beat c. In order for c' to beat c, c' cannot gain any vetoes from the third group of voters. Therefore, for each $S_i \in \mathcal{S}$ at least one $b_j \in S_i$ needs to be added to the election. Thus the q added candidates from B need to correspond to a hitting set of $\mathcal{S}$. ❑

7 Conclusions and Open Problems

We have extended to Condorcet, fallback, and k-veto elections the study of control by replacing voters or candidates initiated by Loreggia et al. (2014; 2015) and pursued later on by Erdélyi, Reger, and Yang (2018). Our complexity results for the associated control problems are summarized in Table 1. We propose to continue the study of electoral control by replacing voters or candidates for other natural voting rules. It would be especially interesting to find a natural voting rule for which the complexity of the standard controls types—in particular, control by adding or deleting voters or candidates—differs from the complexity of control by replacing them.

Admittedly, resistance in terms of NP-hardness—being a *worst-case* measure of complexity only—may not be the last word in wisdom. Indeed, Walsh (2011a; 2011b) and Rothe and Schend (2013) address this issue in electoral control and other manipulative attacks and survey approaches of how to circumvent it. As an ambitious long-term goal, we therefore propose to complement our worst-case complexity analysis by a typical-case analysis of the problems considered here.

Acknowledgements: This work was supported in part by DFG grant RO 1202/14-2.

References

Bartholdi III, J.; Tovey, C.; and Trick, M. 1992. How hard is it to control an election? *Mathematical and Computer Modelling* 16(8/9):27–40.

Baumeister, D., and Rothe, J. 2015. Preference aggregation by voting. In Rothe, J., ed., *Economics and Computation. An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*, Springer Texts in Business and Economics. Springer-Verlag. chapter 4, 197–325.

Baumeister, D.; Erdélyi, G.; Erdélyi, O.; and Rothe, J. 2012. Control in judgment aggregation. In *Proceedings of the 6th European Starting AI Researcher Symposium*, 23–34. IOS Press.

Baumeister, D.; Rothe, J.; and Selker, A. 2015. Complexity of bribery and control for uniform premise-based quota rules under various preference types. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, 432–448. Springer-Verlag *Lecture Notes in Artificial Intelligence #9346*.

Brams, S., and Sanver, R. 2009. Voting systems that combine approval and preference. In Brams, S.; Gehrlein, W.; and Roberts, F., eds., *The Mathematics of Preference, Choice, and Order: Essays in Honor of Peter C. Fishburn*. Springer. 215–237.

Chen, J.; Faliszewski, P.; Niedermeier, R.; and Talmon, N. 2015. Elections with few voters: Candidate control can be easy. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, 2045–2051. AAAI Press.

Erdélyi, G., and Rothe, J. 2010. Control complexity in fallback voting. In *Proceedings of Computing: the 16th Australasian Theory Symposium*, 39–48. Australian Computer Society *Conferences in Research and Practice in Information Technology Series*, vol. 32, no. 8.

Erdélyi, G.; Fellows, M.; Rothe, J.; and Schend, L. 2015a. Control complexity in Bucklin and fallback voting: A theoretical analysis. *Journal of Computer and System Sciences* 81(4):632–660.

Erdélyi, G.; Fellows, M.; Rothe, J.; and Schend, L. 2015b. Control complexity in Bucklin and fallback voting: An experimental analysis. *Journal of Computer and System Sciences* 81(4):661–670.

Erdélyi, G.; Piras, L.; and Rothe, J. 2011. The complexity of voter partition in Bucklin and fallback voting: Solving three open problems. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, 837–844. IFAAMAS.

Erdélyi, G.; Reger, C.; and Yang, Y. 2018. Completing the puzzle: Solving open problems for control in elections. In *Nonarchival website proceedings of the 11th Multidisciplinary Workshop on Advances in Preference Handling*.

Faliszewski, P., and Rothe, J. 2016. Control and bribery in voting. In Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A., eds., *Handbook of Computational Social Choice*. Cambridge University Press. chapter 7, 146–168.

Faliszewski, P.; Hemaspaandra, E.; and Hemaspaandra, L. 2011. Multimode control attacks on elections. *Journal of Artificial Intelligence Research* 40:305–351.

Garey, M., and Johnson, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

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

Lin, A. 2011. The complexity of manipulating k-approval elections. In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence*, 212–218. SciTePress.

Lin, A. 2012. *Solving Hard Problems in Election Systems*. Ph.D. Dissertation, Rochester Institute of Technology, Rochester, NY, USA.

Loreggia, A.; Narodytska, N.; Rossi, F.; Venable, B.; and Walsh, T. 2015. Controlling elections by replacing candidates or votes (extended abstract). In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, 1737–1738. IFAAMAS.

Loreggia, A. 2014. Iterative voting and multi-mode control in preference aggregation. *Intelligenza Artificiale* 8(1):39–51.

Maushagen, C., and Rothe, J. 2016. Complexity of control by partitioning veto and maximin elections and of control by adding candidates to plurality elections. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, 277–285. IOS Press.

Maushagen, C., and Rothe, J. 2017. Complexity of control by partition of voters and of voter groups in veto and other scoring protocols. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, 615–623. IFAAMAS.

Maushagen, C., and Rothe, J. 2018. Complexity of control by partitioning veto elections and of control by adding candidates to plurality elections. *Annals of Mathematics and Artificial Intelligence* 82(4):219–244.

Menton, C. 2013. Normalized range voting broadly resists control. *Theory of Computing Systems* 53(4):507–531.

Rothe, J., and Schend, L. 2013. Challenges to complexity shields that are supposed to protect elections against manipulation and control: A survey. *Annals of Mathematics and Artificial Intelligence* 68(1–3):161–193.

Walsh, T. 2011a. Is computational complexity a barrier to manipulation? *Annals of Mathematics and Artificial Intelligence* 62(1–2):7–26.

Walsh, T. 2011b. Where are the hard manipulation problems? *Journal of Artificial Intelligence Research* 42:1–29.