Autonomous Agents and Multi-Agent Systems (2021) 35:41
https://doi.org/10.1007/s10458-021-09523-9

Towards completing the puzzle: complexity of control by replacing, adding, and deleting candidates or voters

Gábor Erdélyi[1] · Marc Neveling[2] · Christian Reger[3] · Jörg Rothe[2] · Yongjie Yang[4] · Roman Zorn[2]

Accepted: 2 July 2021 / Published online: 29 July 2021

Abstract
We investigate the computational complexity of electoral control in elections. Electoral control describes the scenario where the election chair seeks to alter the outcome of the election by structural changes such as adding, deleting, or replacing either candidates or voters. Such control actions have been studied in the literature for a lot of prominent voting rules. We complement those results by solving several open cases for Copeland$^{\alpha}$, maximin, k-veto, plurality with runoff, veto with runoff, Condorcet, fallback, range voting, and normalized range voting.

Keywords Computational complexity · Electoral control · Copeland · Maximin · Veto · Plurality with runoff · Veto with runoff · Condorcet · Fallback · Range voting · Normalized range voting

1 Introduction

Computational social choice has established itself as a central part in the research and development of multiagent systems and artificial intelligence. Without going into the details here, it is important to note that preference aggregation and voting—and the related scenarios of strategic behavior so as to change the outcome of elections—have many applications in artificial intelligence and, especially, in multiagent systems (e.g., in information extraction [57], planning [15], recommender systems [28], ranking algorithms [14], computational linguistics [53], automated scheduling [32], collaborative filtering [55], etc.). Interestingly, as noted by Hemaspaandra [36, p. 7971], *"At the 2017*

The authors are ordered alphabetically.
This paper merges and extends two preliminary versions that appeared in the proceedings of the *18th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS 2019) [21] and in the proceedings of the *15th International Computer Science Symposium in Russia* (CSR 2020) [50]; the latter paper was also presented at the *16th International Symposium on Artificial Intelligence and Mathematics* (ISAIM 2020) with nonarchival website proceedings.

✉ Yongjie Yang
yyongjiecs@gmail.com

Extended author information available on the last page of the article

AAMAS conference, for example, there were four sessions devoted to Computational Social Choice; no other topic had that many sessions."

Since the seminal work of Bartholdi, Orlin, Tovey, and Trick [5–7], the founders of computational social choice, many strategic voting problems have been proposed and studied from a complexity-theoretic point of view. These strategic voting problems include

- *manipulation* where voters cast their votes strategically;
- *bribery* where an external agent bribes some voters—without exceeding a given budget—so as to change their votes; and
- *electoral control* where an external agent (usually called the chair) tries to alter the outcome of an election by structural changes such as adding, deleting, partitioning, or replacing either candidates or voters.

For a broad overview of these strategic actions and their applications in artificial intelligence and multiagent systems and for a comprehensive survey of related results, we refer to the book chapters by Conitzer and Walsh [12], Faliszewski and Rothe [25], and Baumeister and Rothe [8] and to the comprehensive list of references cited therein.

We will focus on *electoral control*, first and foremost on control by replacing but also on control by adding and by deleting either candidates or voters. There is a long line of research centered on the complexity of control. So, before providing the specific motivation for our results, let us briefly outline the history of research on electoral control, focusing on the particular scenarios we will be concerned with.

Bartholdi, Tovey, and Trick [7] 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 determined which control scenarios these rules are *immune* to (i.e., impossible for the chair to successfully exert control), and in cases where these rules are not immune, 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 [33] 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 elections, they also studied the constructive and destructive control complexity of approval voting.

As surveyed by Faliszewski and Rothe [25] and Baumeister and Rothe [8], plenty of voting rules have been analyzed in terms of their control complexity since then. In addition to the just mentioned results on plurality, Condorcet, and approval voting (and its variants) [7, 9, 16, 19, 33]; the complexity of control in various scenarios has been thoroughly analyzed for Copeland [9, 24]; maximin [23, 45, 47, 61]; k-veto and k-approval [39, 43, 46, 62]; Bucklin and fallback voting [16, 17, 20, 22], range voting and normalized range voting [48], and Schulze voting [49, 54]. Among these voting rules, *fallback voting* (a hybrid system due to Brams and Sanver [10] that combines Bucklin with approval voting) and *normalized range voting* (both will be defined in Sect. 3) are special in that they are the only two natural voting rules with a polynomial-time winner problem that are currently known to have the most resistances to standard control attacks. "Standard control" here refers to

control by adding, deleting, or partitioning either candidates or voters because these are the control types originally introduced by Bartholdi, Tovey, and Trick [7].[1]

On the other hand, the computational complexity of *replacing* either candidates or voters—the control action we mostly focus on—was first studied by Loreggia et al. [40–43]. Replacement control models voting situations in which the number of candidates or voters are predefined and cannot be changed by the chair. For instance, a parliament often consists of a fixed number of seats whose occupants must be replaced if they are removed from their seats. From another viewpoint, the chair might try to veil his or her election tampering via replacement control actions by making sure that the number of participating candidates and voters is the same as before, hoping that the election might appear to be unchanged at first glance. There are also other types of electoral control, such as more natural models of control by partition introduced by Erdélyi, Hemaspaandra, and Hemaspaandra [18], but we will not consider those in this paper.

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 standard control attacks, namely *multimode control*, have been investigated by Faliszewski, Hemaspaandra, and Hemaspaandra [23]. In their model, an external agent is allowed to perform different types of control actions at once such as deleting and/or adding voters and/or candidates. Although some types of multimode control seem to be similar to replacement control, the key difference lies in the tightly coupled control types of replacement control, whereas in multimode control the combined types of standard electoral control can often be handled separately. This leads to the interesting and subtle situation that resistances of voting rules to certain types of standard control do not transfer trivially to related types of replacement control, whereas this indeed can happen for multimode control.

The reader may ask, why do we need yet another paper on the complexity of control? That is, what is the main motivation for the research presented here? Well, the answer is twofold.

First, from a theoretical perspective, it is unsatisfactory that our knowledge about the complexity of control is still incomplete; there are several important voting rules for which we still have some unsolved open cases regarding certain control actions, especially for replacement control. *In this paper, we are filling many of these gaps (see Sect. 2 and, in particular, Table 1 for the details).*

Second, from a practical perspective, a designer of a multiagent system will have to have a careful look at which specific application of voting is planned in his or her system and which strategic scenarios the system will most likely be attacked with. Then, to make

[1] As defined by Bartholdi, Tovey, and Trick [7], for control by partition of either candidates or voters, there is a first round in which the candidates or voters are partitioned into two subgroups which separately elect winners who then may proceed to the final-round election. Hemaspaandra, Hemaspaandra, and Rothe [33] introduced two tie-handling rules, *ties eliminate* and *ties promote*, that determine which of the first-round winners proceed to the final runoff in case of a tie among two or more candidates in any of the two first-round subelections. Further, there are two variants of control by partition of candidates, one *with runoff* (where both subgroups send their winners to the final round) and one *without* (where the winners of one subgroup face *all* candidates of the other in the final round). Hemaspaandra, Hemaspaandra, and Menton [35] showed that certain destructive variants of these problems in fact are the same. In this paper, we will not consider any cases of control by partition, though.

Table 1 Overview of results on the complexity of control by adding, deleting, and replacing either candidates or voters in various voting rules. Our results are in boldface. Previous results [7, 23, 24, 33, 39, 43, 48] are in gray. Entries "NPC" are a shorthand for "NP-completeness" and indicate resistance, "P" vulnerability, and "I" immunity results. The complexity of CCRV for 2-approval —marked by "?"—is still open

(a) Constructive control

	CCAV	CCDV	CCRV	CCAC	CCDC	CCRC
Copeland$^{\alpha}$	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	**NPC**
Maximin	*NPC*	*NPC*	**NPC**	*NPC*	*P*	**NPC**
Plurality	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
2-Approval	*P*	*P*	?	*NPC*	*NPC*	*NPC*
3-Approval	*P*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*
k-Approval, $k \geq 4$	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*
Veto	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
2-Veto	*P*	*P*	**P**	*NPC*	*NPC*	**NPC**
k-Veto, $k \geq 3$	*NPC*	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**
Plurality with runoff	**P**	**P**	**P**	**NPC**	**NPC**	**NPC**
Veto with runoff	**P**	**P**	**P**	**NPC**	**NPC**	**NPC**
Condorcet voting	*NPC*	*NPC*	**NPC**	*I*	*P*	**P**
Fallback voting	*NPC*	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**
Range voting	*NPC*	*NPC*	**NPC**	*I*	*P*	**P**
Normalized range voting	*NPC*	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**

(b) Destructive control

	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
Copeland$^{\alpha}$	*NPC*	*NPC*	**NPC**	*P*	*P*	**P**
Maximin	*NPC*	*NPC*	**NPC**	*P*	*P*	**P**
Plurality	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
2-Approval	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
3-Approval	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
k-Approval, $k \geq 4$	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
Veto	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
2-Veto	*P*	*P*	*P*	*NPC*	*NPC*	**NPC**
k-Veto, $k \geq 3$	*P*	*P*	*P*	*NPC*	*NPC*	**NPC**
Plurality with runoff	**P**	**P**	**P**	**NPC**	**NPC**	**NPC**
Veto with runoff	**P**	**P**	**P**	**NPC**	**NPC**	**NPC**
Condorcet voting	*P*	*P*	**P**	*P*	*I*	**P**
Fallback voting	*P*	*P*	**P**	*NPC*	*NPC*	**NPC**
Range voting	*P*	*P*	**P**	*P*	*I*	**P**
Normalized range voting	*P*	*P*	**P**	*NPC*	*NPC*	**NPC**

a reasonable decision as to which voting rule to choose, the designer will have to know the computational (and other) properties of these strategic (e.g., control) actions against his or her system for the various voting rules. The more complete our knowledge is about the complexity of control scenarios for the most commonly used voting rules, the better will be the designer's decision and the better will be the multiagent system.

Overview of the paper:

Before diving into the technical details of our results, we give an overview of our main contributions in Sect. 2. In Sect. 3, we define the voting rules and control problems to be studied, fix our notation, and give some background on computational complexity. We then study the complexity of various control scenarios for Copeland$^{\alpha}$ in Sect. 4, maximin in Sect. 5, k-veto in Sect. 6, plurality with runoff and veto with runoff in Sect. 7, Condorcet in Sect. 8, fallback in Sect. 9, and for range voting and normalized range voting in Sect. 10. Finally, we conclude in Sect. 11.

2 Our main contributions

In the following, we highlight our main contributions in detail and compare them with the related work to demonstrate how our contributions have improved the state of the art in electoral control. Table 1 gives an overview of previously known and our new results on the complexity of control by replacing, adding, and deleting either candidates or voters for numerous voting rules. For the formal definition of voting rules and control scenarios mentioned and for the notation of control problems, such as CCAV, the reader is referred to Sect. 3.

- Faliszewski et al. [24] and Loreggia [40] investigated the complexity of control in Copeland$^{\alpha}$ elections, leaving open the case of destructive control by replacing voters for any rational α, where $0 \leq \alpha \leq 1$. We settle this open problem.
- Faliszewski, Hemaspaandra, and Hemaspaandra [23] and Maushagen and Rothe [45, 47] investigated the complexity of control in maximin elections but focused on standard control types (i.e., on the cases of constructive and destructive control by adding, deleting, and partitioning either candidates or voters). This leaves the corresponding cases of control by replacing candidates or voters open. We solve these problems. Moreover, we also solve a more general problem called *exact destructive control by adding and deleting candidates*, a special form of multimode control.
- Lin [39] and Loreggia et al. [43] focused on control in k-veto (see also the work of Maushagen and Rothe [46] on control in veto elections). Open cases are constructive control by replacing voters in k-veto elections for $k \geq 2$. We solve these open cases, providing a dichotomy result for k-veto with respect to the values of k.
- The standard control scenarios were studied by Bartholdi, Tovey, and Trick [7] and Hemaspaandra, Hemaspaandra, and Rothe [33] for Condorcet voting, by Erdélyi et al. [16, 17, 20, 22] for fallback elections, and by Menton [48] for range voting and normalized range voting, leaving open for all these rules the cases of constructive and destructive control by replacing either candidates or voters.
- Finally, we investigate the complexity of control for two common voting rules that, somewhat surprisingly, have not been considered yet in the literature, namely plurality with runoff and veto with runoff.

3 Preliminaries

An *election* E is given by a pair $E = (C, V)$, where C is a finite set of *candidates* and V is a finite multiset of *votes*. Voters typically[2] express their preferences over the candidates by 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. When a subset $X \subseteq C$ of candidates occurs in a vote (e.g., $c\ X\ d$ for $X = \{a, b\}$), this means that the candidates in X are ranked in this vote according to a fixed order (e.g., assuming the lexicographic order, $c\ X\ d$ stands for $c\ a\ b\ d$). A *voting rule* (or, more technically, a *voting correspondence*) τ maps each election (C, V) to a subset $W \subseteq C$ of the candidates, called the τ *winners* (or simply the *winners* if τ is clear from the context) of (C, V).

For an election $E = (C, V)$ and two candidates $a, b \in C$, let $N_E(a, b)$ be the number of voters preferring a to b. We drop E from the notation if it is clear from the context. Furthermore, for any set X (e.g., of candidates or voters), let $|X|$ denote the cardinality of X. For ease of exposition, in this paper we exchangeably use the words vote and voter.

Letting $E = (C, V)$ be a given election, we consider the following voting rules.

Copeland$^\alpha$ For each pairwise comparison between any two candidates, say a and b, if $N_E(a, b) > N_E(b, a)$, a receives one point and b zero points. If $N_E(a, b) = N_E(b, a)$, both a and b receive α points, where $\alpha \in [0, 1]$ is a rational number. The *Copeland$^\alpha$ score of any candidate c* is the total number of points c receives from all votes in the election, and all candidates with the highest Copeland$^\alpha$ score win.

Maximin The *maximin score of a candidate $a \in C$* is defined as $\min_{b \in C \setminus \{a\}} N_E(a, b)$, and all candidates with the highest maximin score wins.

***k*-Approval** Each voter gives one point to every candidate in the top-k positions, and all candidates with the highest score win. In particular, 1-approval is often referred to as *plurality voting* in the literature.

***k*-Veto** A candidate gains a point from each vote in which he or she is ranked higher than in the last k positions (i.e., the candidates in the last k positions are vetoed), and all candidates with the highest score win. In particular, 1-veto is simply referred to as *veto*.

Plurality with Runoff (PRun) Each voter only approves of his or her top-ranked candidate. If there is a candidate c who is approved by every voter, then c is the unique winner. Otherwise, this voting rule takes two stages to select the winner. In the first stage, all candidates except the two who receive the, respectively, most and second-most approvals are eliminated from the election. If more than two candidates have the same highest total approvals, a tie-breaking rule

[2] Some voting rules, such as fallback voting, require a different input format to specify votes, as will be explained below.

is applied to select exactly two of them, and if there is one candidate with the most approvals but several candidates with the second-most approvals, a tie-breaking rule is used to select exactly one of those with the second-most approvals. Then the remaining two candidates, say c and d, compete in the second stage (runoff stage). In particular, if $N_E(c,d) > N_E(d,c)$ then c wins; and if $N_E(d,c) > N_E(c,d)$ then d wins. Otherwise, a tie-breaking rule applies to determine the winner between c and d.

Veto with Runoff (VRun) Each voter vetoes exactly the last-ranked candidate. This voting rule is defined similarly to PRun, with a slight difference in the first stage: all candidates except the two candidates who have the least and second-least vetoes are eliminated from the election (again applying a tie-breaking rule if necessary).

Condorcet A *Condorcet winner* is a candidate c who beats all other candidates in pairwise contests, i.e., for each other candidate d, it holds that $N_E(c,d) > N_E(d,c)$. 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 his or her preferences as a subset of candidates $S_v \subseteq C$ that he or she approves of and, in addition, a strict linear ordering of the approved candidates. For instance, if a voter v approves of the candidates $S_v = \{c_1, ..., c_k\} \subseteq C$ and orders them lexicographically, his or 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). For convenience, let $score^0_{(C,V)}(c) = 0$ for every $c \in C$. 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, all candidates with the most level i approvals win.
2. If there is no fallback winner on any level, all candidates with the most approvals win.

Range Voting Instead of a linear order over the m candidates, each voter is associated with a size-m vector $v \in \{0, 1, \ldots, k\}^m$ describing the points the voter gives to each candidate. The number k is the maximum number of points a voter can give to a candidate, i.e., in such a *k-range election*, every voter gives at most k points to a candidate. The *k-range-voting winners* are the candidates with the most points in the given k-range election. 1-range voting is also known as *approval voting*.

Table 2 Special cases of the τ-Constructive-Multimode-Control problem studied in this paper

Problems	Restrictions
Adding voters	$\ell_{\mathrm{AC}} = \ell_{\mathrm{DC}} = \ell_{\mathrm{DV}} = 0, D = \emptyset$
Adding candidates	$\ell_{\mathrm{DC}} = \ell_{\mathrm{AV}} = \ell_{\mathrm{DV}} = 0, W = \emptyset$
Deleting voters	$\ell_{\mathrm{AC}} = \ell_{\mathrm{DC}} = \ell_{\mathrm{AV}} = 0, D = W = \emptyset$
Deleting candidates	$\ell_{\mathrm{AC}} = \ell_{\mathrm{AV}} = \ell_{\mathrm{DV}} = 0, D = W = \emptyset$
Replacing voters	$\lvert V'\rvert = \lvert W'\rvert, \ell_{\mathrm{AV}} = \ell_{\mathrm{DV}}, \ell_{\mathrm{AC}} = \ell_{\mathrm{DC}} = 0, D = \emptyset$
Replacing candidates	$\lvert C'\rvert = \lvert D'\rvert, \ell_{\mathrm{AC}} = \ell_{\mathrm{DC}}, \ell_{\mathrm{AV}} = \ell_{\mathrm{DV}} = 0, W = \emptyset$

Normalized Range Voting Similarly to range voting, each voter is associated with a size-m vector $v \in \{0, 1, \ldots, k\}^m$. Additionally, each voter's vote is normalized to the range of 0 to k in the following way. For each candidate c, let s be the number of points this candidate gains from the voter and $s_{\min}$ and $s_{\max}$ be the minimal and maximal score the voter gives to any candidate. Then the normalized score that v gives to c is $\frac{k(s-s_{\min})}{s_{\max}-s_{\min}}$. Note that if $s_{\max} = s_{\min}$, the voter is indifferent to all candidates and can therefore be ignored. Again, the *k-normalized-range-voting winners* are the candidates with the most normalized points in the given k-range election.

We study various control problems that can be considered as special cases of the following problem [23], which is defined for a given voting rule τ.

τ-Constructive-Multimode-Control

Input: An election $(C \cup D, V \cup W)$ with a set C of (registered) candidates,[3] a set D of as yet unregistered candidates, a list V of registered voters, a list W of as yet unregistered voters, a distinguished candidate $c \in C$, and four nonnegative integers ℓ_{AV}, ℓ_{DV}, ℓ_{AC}, and ℓ_{DC}, with $\ell_{\mathrm{AV}} \leq |W|$, $\ell_{\mathrm{DV}} \leq |V|$, $\ell_{\mathrm{AC}} \leq |D|$, and $\ell_{\mathrm{DC}} \leq |C|$.

Question: Are there $V' \subseteq V, W' \subseteq W, C' \subseteq C \setminus \{c\}$, and $D' \subseteq D$ such that $|V'| \leq \ell_{\mathrm{DV}}, |W'| \leq \ell_{\mathrm{AV}}$, $|C'| \leq \ell_{\mathrm{DC}}, |D'| \leq \ell_{\mathrm{AC}}$, and c is a τ winner of the election $((C \setminus C') \cup D', (V \setminus V') \cup W')$?

[3] We may sometimes omit mentioning explicitly that these candidates are registered.

In τ-Destructive-Multimode-Control, we ask whether there exist subsets V', W', C', and D' as in the above definition such that c is *not* a τ winner in $((C \setminus C') \cup D', (V \setminus V') \cup W')$.

We will study several special cases or restricted versions of multimode control, such as adding, deleting, or replacing either candidates or voters. Table 2 gives an overview of the restrictions compared to the general multimode control problem.

Throughout the paper, we will use a four-letter code to denote our problems. The first two characters CC/DC stand for *constructive/destructive control*, the third character A/D/R stands for *adding/deleting/replacing*, and the last one V/C for *voters/candidates*. For example, DCRV stands for *destructive control by replacing voters*. For simplicity, in each problem in the above table, we use ℓ to denote the integer(s) in the input that is not necessarily required to be 0. For example, when considering CCRV, we use ℓ to denote $\ell_{\mathrm{AV}} = \ell_{\mathrm{DV}}$.

As mentioned in the introduction, since the seminal work of Bartholdi, Tovey, and Trick [7] control by *adding* and *deleting* candidates or voters has been extensively studied in the literature (see, e.g., [11, 17, 34, 44, 49, 60, 62]). However, the complexity of control by *replacing* candidates or voters has been introduced and studied just recently by Loreggia et al. [40–43].

We remark that our proofs are based on the nonunique-winner model but can be modified to work for the unique-winner model of the control problems as well.[3]

We assume the reader to be familiar with the basics of complexity theory, such as the complexity classes P and NP and the notions of NP-hardness and NP-completeness under (polynomial-time many-one) reductions. We refer to Tovey's tutorial [58] for a concise introduction to complexity theory and to the books by Arora and Barak [2], Garey and Johnson [27], and Rothe [56] for more comprehensive discussions.

We call a voting rule *immune* to a type of control if it is never possible for the chair to reach his or 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 easily seen to be in NP, so any resistance result immediately implies NP-completeness, and we only provide the NP-hardness proofs since membership of these problems in NP is easy to check. Our NP-hardness results are mainly based on reductions from the Restricted-Exact-Cover-By-3-Sets (RX3C) problem [29] and the Hitting-Set problem [37]:

Restricted-Exact-Cover-By-3-Sets (RX3C)

Input: A set $U = \{u_1, \ldots, u_{3\kappa}\}$ and a collection $\mathscr{S} = \{S_1, \ldots, S_{3\kappa}\}$ of 3-element subsets of U such that each $u \in U$ occurs in exactly three subsets $S \in \mathscr{S}$.

Question: Does $\mathscr{S}$ contain an exact 3-set cover for U, i.e., a subcollection $\mathscr{S}' \subseteq \mathscr{S}$ such that every element of U occurs in exactly one member of $\mathscr{S}'$?

If we do not request every $u \in U$ to occur in exactly three elements of $\mathscr{S}$ in the RX3C problem, we obtain the generalized X3C problem.

Hitting-Set

Input: A set $U = \{u_1, \ldots, u_s\}$ with $s \geq 1$, a family $\mathscr{S} = \{S_1, \ldots, S_t\}$ of nonempty subsets $S_i \subseteq U$, and an integer κ with $1 \leq \kappa \leq s$.

Question: Is there a subset $U' \subseteq U, |U'| \leq \kappa$, such that each $S_i \in \mathscr{S}$ is *hit* by U' (i.e., $S_i \cap U' \neq \emptyset$ for all $S_i \in \mathscr{S}$)?

Note further that all voting rules considered here are susceptible to the control scenarios we study. Since the corresponding proofs can be easily obtained by appropriate examples, we will omit them in most cases. The only exceptions are Condorcet and range voting: While among the voting rules we consider these two are the only ones that are immune to some of the standard control scenarios (namely, to constructive control by

[3] In the *nonunique-winner model*, for a constructive (respectively, destructive) control action to be successful, it is enough to make the distinguished candidate c a winner, possibly among others, of the resulting election (respectively, it must be ensured that c is not even a winner), whereas in the *unique-winner model*, a constructive (respectively, destructive) control action is considered to be successful only when c alone wins (respectively, it is enough to ensure that c is not the only winner).

Table 3 Complexity of control for Copeland$^\alpha$. Our results are in boldface. "NPC" stands for "NP-complete" and "P" stands for "polynomial-time solvable"

CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
NPC	*NPC*	*NPC*	*NPC*	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**	*P*	*P*	**P**

adding candidates [7, 48] and to destructive control by deleting candidates [33, 48]), we will explicitly show that susceptibility holds in these control scenarios for Condorcet (see Example 1) and range voting (see Example 2).

Assuming that the reader is familiar with graph theory (see also the books by Bang-Jensen and Gutin [4] and West [59]), we will in some proofs make use of the following problems to show membership in P.

INTEGRAL-MINIMUM-COST-FLOW (IMCF)

Input: A network $G = (V, E)$, capacity functions $b_\alpha, b_\beta : E \to \mathbb{N}_0$, a source vertex $x \in V$, a sink vertex $y \in V \setminus \{x\}$, a cost function $g : E \to N_0$, and an integer r.

Task: Find a minimum cost flow from x to y of value r. Recall that a flow f is a function assigning to each arc $(u, v) \in E$ an integer number $f(u, v)$ such that (1) $b_\alpha(u, v) \leq f(u, v) \leq b_\beta(u, v)$; and (2) for every node v except x and y, it holds that $\sum_{(u,v)\in E} f(u, v) = \sum_{(v,u)\in E} f(v, u)$.[4] The cost of a flow f is $\sum_{(u,v)\in E} f(u, v) \cdot g(u, v)$, and the value of f is $\sum_{(x,v)\in E} f(x, v)$.

In the above definitions, b_α and b_β are called the *lower-bound capacity* and the *upper-bound capacity*, respectively. The IMCF problem is well-known to be polynomial-time solvable [1].

b-EDGE-COVER (b-EC)

Input: An undirected multigraph $G = (V, E)$ without loops, two capacity functions $b_\alpha, b_\beta : V \to \mathbb{N}_0$, and an integer r.

Question: Is there a b-edge cover in G of size at most r, i.e., a subset $E' \subseteq E$ of at most r edges such that each node $v \in V$ is incident to at least $b_\alpha(v)$ and at most $b_\beta(v)$ edges in E'?

The b-EC problem is also known to be polynomial-time solvable [26, 30].

4 Copeland$^\alpha$ voting

We start by completing our knowledge on control complexity in Copeland$^\alpha$ elections. Previously, Faliszewski et al. [24] and Loreggia [40] investigated the complexity of control in Copeland$^\alpha$ elections, leaving open the cases of destructive control by replacing voters and of constructive and destructive control by replacing candidates. In this section, we fill the gaps. We refer to Table 3 for a summary of our results in this section.

[4] For simplicity, we write $b_\alpha(u, v)$ for $b_\alpha((u, v))$, $b_\beta(u, v)$ for $b_\beta((u, v))$, and $g(u, v)$ for $g((u, v))$ throughout this paper.

Definition 1 (Lang, Maudet, and Polukarov [38]) A voting rule satisfies *Insensitivity to Bottom-ranked Candidates (IBC)* if for any election with at least two candidates, the winners do not change after deleting a subset of candidates who are ranked after all other candidates in all votes.

Note that both Copeland$^\alpha$ and maximin satisfy IBC. Loreggia et al. [42, 43] established the following relationship between CCRC and CCDC, and between DCRC and DCDC.

Lemma 1 (Loreggia et al. [42, 43]) *Let τ be a voting rule satisfying IBC. Then τ-CCRC is* NP*-hard if τ-CCDC is* NP*-hard, and τ-DCRC is* NP*-hard if τ-DCDC is* NP*-hard.*

By Lemma 1 and the facts that Copeland$^\alpha$ satisfies IBC and that, as shown by Faliszewski et al. [24], Copeland$^\alpha$-CCDC is NP-hard for any rational α with $0 \leq \alpha \leq 1$, we have the following result.

Corollary 1 *For any rational α with $0 \leq \alpha \leq 1$, Copeland$^\alpha$-CCRC is* NP*-complete.*

However, for each rational α with $0 \leq \alpha \leq 1$, Copeland$^\alpha$-DCDC is *not* NP-hard but in P [24], so Lemma 1 does not imply NP-hardness of Copeland$^\alpha$-DCRC. In fact, we now show that this problem can be solved in polynomial time.

Theorem 1 *For any rational α with $0 \leq \alpha \leq 1$, Copeland$^\alpha$-DCRC is in* P*.*

Proof To show membership in P, we will provide an algorithm that runs in polynomial time. Given a Copeland$^\alpha$-DCRC instance $((C \cup D, V), c, \ell)$, we first check the trivial case, and immediately accept if c is already not winning the election (C, V). Otherwise, for any two candidates $c_1, c_2 \in C \cup D$, let $\text{Score}(c_1, c_2)$ be the number of points c_1 receives by c_2's presence in the election (i.e., $\text{Score}(c_1, c_2) = 1$ if $N_{(C\cup D,V)}(c_1, c_2) > N_{(C\cup D,V)}(c_2, c_1)$, $\text{Score}(c_1, c_2) = \alpha$ if $N_{(C\cup D,V)}(c_1, c_2) = N_{(C\cup D,V)}(c_2, c_1)$, and $\text{Score}(c_1, c_2) = 0$ otherwise).[5] We now try to find a candidate $d \in (C \cup D) \setminus \{c\}$ and an integer ℓ' with $1 \leq \ell' \leq \ell$ so that d beats c by replacing ℓ' candidates. For a pair (d, ℓ'), we can check if this is possible in polynomial time in the following way. Firstly, we compute $\text{Score}(c, e)$ and $\text{Score}(d, e)$ for every $e \in (C \cup D) \setminus \{c, d\}$. Then we sort $C \setminus \{c, d\}$ in decreasing order according to $\text{Score}(c, e) - \text{Score}(d, e)$ for each candidate $e \in C \setminus \{c, d\}$ and let $C' \subseteq C \setminus \{c, d\}$ contain the first ℓ' candidates according to this ordering. Furthermore, we sort $D \setminus \{d\}$ in decreasing order according to $\text{Score}(d, e) - \text{Score}(c, e)$ and let $D' \subseteq D \setminus \{d\}$ contain the first ℓ' candidates according to this ordering if $d \notin D$ and the first $\ell' - 1$ candidates according to this ordering if $d \in D$. We then check if c is not winning in $((C \setminus C') \cup D' \cup \{d\}, V)$.

Correctness of the algorithm follows from the fact that we iterate over all possible candidates that can prevent c from winning and all possible numbers of replacements we may need to this end, and then check whether we can be successful by adding and deleting the most optimal candidates in regards to how they affect the points balance of c and the candidate that should beat c after this replacement.

To see that the above algorithm runs in polynomial time, note that we can iterate over all pairs of candidates and replacements in $O(|C \cup D|\ell)$ time and checking whether a pair

[5] Note that the value of $\text{Score}(c_1, c_2)$ does not depend on any other candidates in the election.

is successful takes $O(|C|\log(|C|) + |D|\log(|D|))$ time for sorting and choosing the subsets and polynomial time for winner determination. □

It remains to handle the case of destructive control by replacing voters. We solve it in the following theorem.

Theorem 2 *For any rational α with $0 \leq \alpha \leq 1$, Copeland$^\alpha$-DCRV is* NP*-complete.*

Proof Our proof is a slight modification of the proof of Theorem 4.17 (showing that for every rational number α such that $0 \leq \alpha \leq 1$, Copeland$^\alpha$-CCAV is NP-complete) given by Faliszewski et al. [24], with the only difference that there are a number of new registered votes. In particular, from an instance $(U, \mathscr{S})$ of the RX3C problem, it is shown by Faliszewski et al. [24] that an instance of CCAV with the following property can be constructed in polynomial time.[6] Let $|U| = |\mathscr{S}| = 3\kappa$. The candidate set is

$$C = U \cup \{p, r, s\} \cup D,$$

where D is a set of t padding candidates with t a sufficiently large integer but bounded by a polynomial in κ (e.g., $t = 9(\kappa + 1)^3$). The multiset V of registered votes are constructed so that, with respect to these registered votes, the Copeland$^\alpha$ scores of p is t, of r is $t + 3\kappa$, and of every other candidate is at most $t - 1$. Moreover, it holds that

- $N_{(C,V)}(s, p) - N_{(C,V)}(p, s) = \kappa - 1$,
- $N_{(C,V)}(r, u) - N_{(C,V)}(u, r) = \kappa - 3$ for every $u \in U$, and
- $|N_{(C,V)}(c, c') - N_{(C,V)}(c', c)| \geq \kappa + 1$ for all other pairs of candidates c and c' in C. ($|N_{(C,V)}(c, c') - N_{(C,V)}(c', c)|$ is the absolute value of $N_{(C,V)}(c, c') - N_{(C,V)}(c', c)$.)

We refer to [24] for the details of how these votes are created. In addition to the above registered votes, we add the following registered votes. First, for every two candidates $c, c' \in C$ such that $N_{(C,V)}(c, c') - N_{(C,V)}(c', c) \geq \kappa + 1$, we add 2κ registered votes, among which κ of them are of the form $c\ c'\ C \setminus \{c, c'\}$ and the other κ of them are of the form $\overrightarrow{C \setminus \{c, c'\}}\ c\ c'$, where $\overrightarrow{C \setminus \{c, c'\}}$ is the reversal of $C \setminus \{c, c'\}$. Let V_1 be the multiset of the above newly added votes. Then we add a multiset V_2 of κ votes, each of which ranks r in the top, ranks p in the last place, and ranks s just before p. (Other candidates are ranked arbitrarily between r and s.) For notational brevity, let us redefine $V := V \cup V_1 \cup V_2$ as the multiset of all registered votes hereinafter in the proof. Then it is fairly easy to check that the following conditions hold.

- The Copeland$^\alpha$ scores of all candidates remain the same as before the creation of $V_1 \cup V_2$;
- $N_{(C,V)}(s, p) - N_{(C,V)}(p, s) = 2\kappa - 1$;
- $N_{(C,V)}(r, u) - N_{(C,V)}(u, r) = 2\kappa - 3$ for every $u \in U$; and
- $|N_{(C,V)}(c, c') - N_{(C,V)}(c', c)| \geq 2\kappa + 1$ holds for all other pairs of candidates c and c' not specified above.

[6] The reduction in [24] is in fact from the X3C problem, which is a generalization of RX3C where the restriction that every $u \in U$ occurs in exactly three elements of $\mathscr{S}$ is dropped.

Table 4 Complexity of control for maximin. Our results are in boldface. "NPC" stands for "NP-complete" and "P" stands for "polynomial-time solvable"

CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
NPC	*NPC*	**NPC**	*NPC*	*P*	**NPC**	*NPC*	*NPC*	**NPC**	*P*	*P*	**P**

The unregistered votes are constructed according to $\mathscr{S}$. Precisely, for every $S \in \mathscr{S}$, there is an unregistered vote with the following preference:

$$p\ (U \setminus S)\ r\ S\ (C \setminus (\{p, r, s\} \cup U))\ s.$$

Let W denote the set of all unregistered votes. Additionally, we set $\ell = \kappa$. Finally, we let r be the distinguished candidate (who is the current winner).

We move on to the proof for the equivalence of the two instances.

($\Rightarrow$) Assume that U admits an exact set cover $\mathscr{S}' \subseteq \mathscr{S}$. Let $W' \subseteq W$ be the set of unregistered votes corresponding to $\mathscr{S}'$. We claim that after replacing V_2 with W', r is not a winner anymore. Let $E = (C, V \setminus V_2 \cup W')$. Observe that if $|N_{(C,V)}(c, c') - N_{(C,V)}(c', c)| > 2\kappa + 1$, then c still beats c' in E, as we replace at most κ votes. As $\mathscr{S}'$ is an exact set cover of U, for every $u \in U$, there are exactly $\kappa - 1$ votes in W' which rank u above r. In addition, as $N_{(C,V)}(r, u) = 2\kappa - 3$ holds for every $u \in U$ and all votes in V_2 rank r in the first place, we know that r is beaten by all candidates in U in the election E. So, the Copeland$^\alpha$ score of r decreases to t in E. Moreover, as all votes in V_2 rank s above p, all votes in W' rank p in the top, and $N_{(C,V)}(s, p) - N_{(C,V)}(p, s) = 2\kappa - 1$, we have that $N_E(p, s) - N_E(s, p) = 1$, i.e., in the election E the candidate p beats s. Therefore, the Copeland$^\alpha$ score of p in E increases to $t + 1$. Clearly, r is no more a winner in E.

($\Leftarrow$) Assume that there are $V' \subseteq V$ and $W' \subseteq W$ such that $|V'| = |W'| \leq \kappa$, and r is not a winner in the election $E = (C, V \setminus V' \cup W')$. As pointed out above, if $N_{(C,V)}(c, c') - N_{(C,V)}(c', c) \geq 2\kappa + 1$, then c still beats c' after replacing at most κ votes. This means that replacing at most κ votes can only change the Copeland$^\alpha$ scores of p, s, and r (see the above conditions). More importantly, between p and s, as all unregistered votes rank s in the last place, replacing at most κ votes does not increase the score of s. Moreover, as $|N_{(C,V)}(r, c') - N_{(C,V)}(c', r)| \geq 2\kappa + 1$ for all other candidates $c' \in C \setminus U$, replacing at most κ votes can only change the head-to-head comparisons between r and candidates in U. This implies that in the election E, r has Copeland$^\alpha$ score at least t. Therefore, we know that p is the only candidate that prevents r from winning in E. Then, as $|N_{(C,V)}(p, c) - N_{(C,V)}(c, p)| \geq 2\kappa - 1$ for all candidates $c \in C \setminus \{p, s\}$, the Copeland$^\alpha$ score of p in E can be at most $t + 1$. This implies that the Copeland$^\alpha$ score of r in E is exactly t. As the comparisons between r and any of the other candidates in $C \setminus U$ do not change by replacing at most κ votes, this is possible only when r is beaten by everyone in U in the election E. This means that for every $u \in U$, there are at least $\kappa - 1$ votes in W' which rank u above r. Due to the construction of the unregistered votes, for each $S \in \mathscr{S}$ that corresponds to an unregistered vote ranking u above r, it holds that $u \notin S$. As this holds for all $u \in U$ and W' contains at most κ votes, we can conclude that the subcollection of $\mathscr{S}$ corresponding to W' is an exact set cover of U. □

Table 5 Head-to-head comparisons of candidates with respect to the registered votes in the proof of Theorem 3. * means that the value does not have any impact on the correctness of the reduction

	c	d	$u \in U$	maximin score
c	–	2κ	2κ	2κ
d	$3\kappa + 1$	–	$4\kappa + 1$	$3\kappa + 1$
$u' \in U$	$3\kappa + 1$	κ	*	$\leq \kappa$

5 Maximin voting

Let us now turn to maximin voting. Faliszewski, Hemaspaandra, and Hemaspaandra [23] have already investigated the complexity of constructive and destructive control by adding and deleting either candidates or voters. Maushagen and Rothe [45, 47] settled all cases of constructive and destructive control by partitioning either candidates or voters. We will complete the picture on control in maximin elections by providing results on constructive and destructive control by replacing either candidates or voters. Our results in this section are summarized in Table 4.

It is known that constructive control by deleting candidates for maximin is polynomial-time solvable [23]. Hence, assuming P $\neq$ NP, Lemma 1 cannot be used to obtain NP-hardness of Maximin-CCRC. However, as stated below, Loreggia [42] introduced another useful lemma.

Definition 2 A voting rule is said to be *unanimous* if whenever the same candidate is ranked in the top position in all votes, this candidate wins.

Lemma 2 (Loreggia [42]) *Let τ be an unanimous voting rule that satisfies IBC. If τ-CCAC is* NP*-hard, then τ-CCRC is* NP*-hard.*

Due to this lemma and the facts that (1) maximin is unanimous; (2) maximin satisfies IBC; and (3) Maximin-CCAC is NP-complete [23], we have

Corollary 2 *Maximin-CCRC is* NP*-complete.*

The following theorem handles constructive and destructive control by replacing voters. Our proof is a modification of the proof of constructive control by adding voters in maximin [23]. In the following, for two subsets A and B of candidates and a linear order over candidates, $A\ B$ means that $a\ b$ for every $a \in A$ and $b \in B$.

Theorem 3 *Maximin-CCRV and Maximin-DCRV are* NP*-complete.*

Proof We start with the constructive case. Let $(U, \mathscr{S})$ be a given RX3C instance such that $|U| = |\mathscr{S}| = 3\kappa$. We construct the following Maximin-CCRV instance. Let the set of candidates be $C = U \cup \{c, d\}$ such that $\{c, d\} \cap U = \emptyset$. The distinguished candidate is c. The registered votes are as follows:

- there are $3\kappa + 1$ votes of the form $d\ U\ c$;
- there are κ votes of the form $c\ U\ d$; and

- there are κ votes of the form $c\ d\ U$.

Let V denote the multiset of the above $5\kappa + 1$ registered votes. The head-to-head comparisons of candidates (i.e., $|N_{(C,V)}(c, c')|$ for all $c, c' \in C$) and their maximin scores with respect to the registered votes are summarized in Table 5.

Moreover, for each $S \in \mathscr{S}$, we create an unregistered vote in W of the form

$$(U \setminus S)\ c\ S\ d.$$

We use $v(S)$ to denote this vote. Finally, we set $\ell = \kappa$, i.e., we are allowed to replace at most κ voters.

The above Maximin-CCRV instance clearly can be constructed in polynomial time. We claim that we can make c the winner of the election by replacing up to κ voters if and only if $\mathscr{S}$ contains an exact set cover of U.

($\Rightarrow$) Assume that U admits an exact set cover $\mathscr{S}' \subseteq \mathscr{S}$. Let $W' = \{v(S) \mid S \in \mathscr{S}'\}$ be the set of the unregistered votes corresponding to this exact set cover. Clearly, $|W'| = |\mathscr{S}'| = \kappa$. Let V' be a multiset of κ registered votes of the form $d\ U\ c$. We claim that c becomes a winner in the election $E' = (C, (V \setminus V') \cup W')$. Let us now analyze the maximin scores of the candidates in E'. First, as all votes in W' rank c above d, and all votes in V' rank c in the last position, it holds that $N_{E'}(c, d) = 2\kappa - 0 + \kappa = 3\kappa$. As $\mathscr{S}'$ is an exact set cover of U, for every candidate $u \in U$ there is exactly one vote, namely, the vote $v(S)$ such that $u \in S$, which ranks c above u and is contained in V'. In addition, as all votes in V' rank c in the end, we know that $N_{E'}(c, u) = 2\kappa + 1$ for every $u \in U$. So, the maximin score of c in the election E' increases from 2κ to $2\kappa + 1$. Now we start the analysis for the candidate d. As all votes in W' rank d in the last position and all votes in V' rank d in the first position, the maximin score of d in E' decreases from $3\kappa + 1$ to $2\kappa + 1$. As the maximin score of every candidate $u \in U$ is at most κ with respect to V, and we are allowed to replace at most κ votes, the maximin score of u in E' can be at most 2κ. In summary, c and d are the only two candidates having the maximum maximin score in E', and hence c is a winner in E'.

($\Leftarrow$) Assume that there is a subset $V' \subseteq V$ and a subset $W' \subseteq W$ such that $|V'| = |W'| \leq \kappa$ and c wins the election $(C, (V \setminus V') \cup W')$. Let $\hat{E} = (C, (V \setminus V') \cup W')$, and let $\mathscr{S}' = \{S \in \mathscr{S} \mid v(S) \in W'\}$. An important observation is that the maximin score of c in $\hat{E}$ can be at most $2\kappa + 1$. In fact, no matter which up to κ unregistered votes are included in W', there is at least one candidate $u \in U$ such that there is at most one unregistered vote in W' which ranks c above u, implying that $N_{\hat{E}}(c, u) \leq 2\kappa + 1$. From this observation, we know that V' must consist of exactly κ votes and, moreover, all votes in V' must rank d above c, since otherwise d would have maximin score at least $3\kappa + 1 - (\kappa - 1) = 2\kappa + 2$ in $\hat{E}$, contradicting that c is a winner in $\hat{E}$. This means that V' consists of exactly κ registered votes of the form $d\ U\ c$. Now the maximin score of d in $\hat{E}$ is determined as $3\kappa + 1 - \kappa = 2\kappa + 1$. We claim that $\mathscr{S}'$ is an exact set cover of U. For the sake of contradiction, assume that this is not the case. Then there is a candidate $u \in U$ such that none of the sets in $\mathscr{S}'$ contains u. In light of the above construction of the unregistered votes, all the κ votes in W' rank this particular candidate u above c, resulting in the maximin score of c in $\hat{E}$ being at most 2κ, contradicting that c is a winner in E'.

The destructive version works identically, except that the first group of votes (i.e., votes of the type $d\ U\ c$) consists of 3κ registered votes and the distinguished candidate is d. In this case, one can check that, similarly to the analysis in the above ($\Rightarrow$) direction, after replacing κ registered votes of the form $d\ U\ c$ with κ unregistered votes corresponding to

an exact set cover of U, the maximin scores of c and d are, respectively, $2\kappa + 1$ and 2κ, leading to d not being a winner anymore. For the proof of the other direction, one observes that the maximin score of d, after replacing at most κ votes from V and by as many votes from W, is at least $3\kappa - \kappa = 2\kappa$, and the maximin score of every $u \in U$ can be at most 2κ. This means that c is the only candidate that may have maximin score at least $2\kappa + 1$ in the final election. Analogously to the analysis in the above ($\Leftarrow$) direction, we can show that the candidate c achieves the maximin score $2\kappa + 1$ if and only if there exists a set of κ unregistered votes corresponding to an exact set cover of U. □

It remains to show the complexity of destructive control by replacing candidates for maximin. In contrast to the NP-hardness results for the other replacing cases, we show that Maximin-DCRC is polynomial-time solvable. In fact, we show P membership of a more general problem called τ-Exact-Destructive-Control-by-Adding-and-Deleting-Candidates, denoted by τ-EDCAC+DC, where τ is a voting rule. In particular, this problem is a variant of τ-Destructive-Multimode-Control, where $\ell_{\mathrm{AV}} = \ell_{\mathrm{DV}} = 0, W = \emptyset$. Moreover, it must hold that in the solution $|C'| = \ell_{\mathrm{DC}}$ and $|D'| = \ell_{\mathrm{AC}}$ (i.e., the chair deletes *exactly* ℓ_{DC} candidates and adds *exactly* ℓ_{AC} candidates). Note that the number of candidates added and the number of candidates deleted do not have to be the same.

Theorem 4 *Maximin-EDCAC+DC is in* P.

Proof Our input is a Maximin-EDCAC+DC instance as defined above. Suppose that the chair adds exactly ℓ_{AC} candidates from D and deletes exactly ℓ_{DC} candidates from C. Note that $\ell_{\mathrm{DC}} < |C|$ since the chair must not delete the distinguished candidate c. Our algorithm works as follows. It checks if there is a pivotal candidate $c' \neq c$ that beats c in the final election. In case c has maximin score at most k for some integer k in the final election, there exists some candidate $d \in (C \cup D) \setminus \{c\}$, not necessarily different from c' with $N(c, d) \leq k$. Our algorithm checks whether there is a final election including c, c', and d, the candidate c has maximin score at most k, and c' has maximin score at least $k + 1$, where $k \in \{0, 1, \ldots, |V| - 1\}$. Note that we may restrict ourselves to values $k \leq \lceil |V|/2 \rceil - 1$. Otherwise, c does not lose any pairwise comparison and is a weak Condorcet winner and thus a maximin winner.

In more detail, the algorithm first tries to find the candidate $c' \in (C \cup D) \setminus \{c\}$ and the threshold score k as discussed above, and then proceeds with the following steps.

1. Let $D(c') = \{d \in (C \cup D) \setminus \{c\} : N(c, d) \leq k \wedge (c' = d \vee N(c', d) > k)\}$. If $D(c') = \emptyset$ or $N(c', c) \leq k$, we immediately reject for the pair (c', k). Otherwise, we try to find a candidate $d \in D(c')$ (not necessarily different from c'). The candidate d has the function to fix the score of c below or equal to k. In order to keep c''s score above the score of c, it must hold either $c' = d$ or $N(c', d) > k$.[7] We go to the next step.
2. Check whether $\ell_{\mathrm{DC}} \leq |C| - 1 - |C \cap \{c', d\}|$ and $\ell_{\mathrm{AC}} \geq |D \cap \{c', d\}|$. If this is the case, proceed with the next step. Otherwise, we reject because there is no way for the chair to keep both c' and d in (or to add them to) the final election.

[7] Note that if the maximin score of c is less than k, the candidate c' can also beat c with maximin score k, but this case is captured by another pair (c', k).

3. Let $C_1 = \{c'' \in C \setminus \{c, c', d\} : N(c', c'') \leq k\}$. The candidates in C_1 must all be deleted in order to keep the maximin score of c' higher than k. If $|C_1| > \ell_{\text{DC}}$, we discard this subcase and try the next triple (c', k, d). Otherwise, the chair deletes all candidates in C_1 and arbitrary other candidates in $C \setminus \{c, c', d\}$ such that exactly ℓ_{DC} candidates have been deleted. We go to the next step.
4. Let $D_1 = \{a \in D \setminus \{c', d\} : N(c', a) > k\}$. Candidates in D_1 are the only candidates which may be added and the score of c' does not decrease. Hence, if $|D_1| < \ell_{\text{AC}} - |D \cap \{c', d\}|$, we reject for the triple (c', k, d) since the chair must add some candidates leading to a lower score than $k + 1$ for c'. Otherwise, we accept.

If the given instance is a YES-instance, at least one such triple (c', k, d) must lead to the algorithm accepting it. However, if we are given a NO-instance, the algorithm must reject. Finally, the algorithm runs in polynomial time because there are polynomially many triples to check and each of them can be done in polynomial time as described above. □

Note that Maximin-DCRC is polynomial-time Turing-reducible to Maximin-EDCAC+DC. Then, from Theorem 4 we obtain the following result.

Corollary 3 *Maximin-DCRC is in* P.

Theorem 4 generalizes the polynomial-time solvability results for Maximin-DCAC and Maximin-DCDC obtained by Faliszewski et al. [23]. We also point out that Faliszewski, Hemaspaandra, and Hemaspaandra [23] showed that Maximin-CCAC$_{\text{u}}$+DC is polynomial-time solvable, where the subscript u refers to control by adding an *unlimited* number of candidates, as originally defined by Bartholdi, Tovey, and Trick [7]: In this case, the chair is allowed to add as many unregistered candidates as desired but can only delete a limited number of candidates.

6 *k*-veto

Turning now to k-veto and starting with control by replacing voters, it is known that Veto-CCRV and k-Veto-DCRV for all possible k are polynomial-time solvable [43], which leaves open the complexity of k-Veto-CCRV for $k \geq 2$. We complement these results by showing that 2-Veto-CCRV is polynomial-time solvable and k-Veto-CCRV is NP-complete for $k \geq 3$, achieving a dichotomy result for constructive control by replacing voters in k-veto with respect to the values of k. Our results in this section are summarized in Table 6.

As a notation, let V^c (W^c) be the set consisting of all voters in V (W) vetoing c, and define $V^{\neg c} = V \setminus V^c$ ($W^{\neg c} = W \setminus W^c$).

Theorem 5 2-*Veto-CCRV is in* P.

Proof Let $(C, V \cup W)$, ℓ, and $c \in C$ be the components of a given 2-Veto-CCRV instance, as described in Sect. 3. Recall that c is the distinguished candidate in the input. Our algorithm distinguishes the following cases:

Case 1: $|V^c| \leq \min(\ell, |W| - |W^c|)$.

Table 6 Complexity of control for k-veto. Our results are in boldface. "NPC" stands for "NP-complete" and "P" stands for "polynomial-time solvable"

	CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
$k = 1$	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*	*P*	*P*	*P*	*NPC*	*NPC*	*NPC*
$k = 2$	*P*	*P*	**P**	*NPC*	*NPC*	**NPC**	*P*	*P*	*P*	*NPC*	*NPC*	**NPC**
$k \geq 3$	*NPC*	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**	*P*	*P*	*P*	*NPC*	*NPC*	**NPC**

In this case, the algorithm returns "YES" since c can be made a winner with zero vetoes by replacing all registered votes vetoing c with the same number of unregistered votes not vetoing c.

Case 2: $|W| - |W^c| \leq \min(\ell, |V^c|)$.

In this case, the optimal choice for the chair is to replace $|W| - |W^c|$ voters in V vetoing c by the same number of voters from W not vetoing c. Hence, all votes in $W^{\neg c}$ are ensured in the final election. In addition, all votes in $V^{\neg c}$ are also in the final election, as none of these votes needs to be exchanged in an optimal solution. However, the chair possibly needs to exchange further $\ell - |W| + |W^c|$ V-voters vetoing c by the same number of W-voters vetoing c. Anyway, c has exactly

$$v_c = |V^c| - (|W| - |W^c|) = |(V \cup W)^c| - |W|$$

vetoes in the final election. Due to these observations, the question is equivalent to searching for no more than v_c voters in $V^c \cup W^c$ that shall belong to the final election such that at least $\max(0, |V^c| - \ell)$ and at most $|V^c| - |W| + |W^c|$ among them belong to V^c. We sequentially check for the exact number ℓ', where

$$\max(0, |V^c| - \ell) \leq \ell' \leq |V^c| - |W| + |W^c|,$$

of V-voters that are kept in the final election. This implies that we keep exactly $v_c - \ell'$ votes from W^c in the final election. Clearly, if the given instance falls into this case and is a YES-instance, at least one of these checked numbers leads to a YES answer.

In the following, we transform the instance into an equivalent b-EC instance in polynomial time, thus providing a reduction from 2-Veto-CCRV to b-EC.

For each candidate $d \in C \setminus \{c\}$, we create a vertex d. In addition, we create two vertices c_V and c_W representing vetoes that nondistinguished candidates receive from voters in V or W vetoing c, respectively. Each voter in V^c (W^c) vetoing some candidate $d \in C \setminus \{c\}$ and c yields an edge between d and c_V (c_W). The capacities are as follows:

- $b_\alpha(c_V) = b_\beta(c_V) = \ell'$. These capacities ensure that exactly ℓ' votes from V^c are kept in the final election.
- $b_\alpha(c_W) = b_\beta(c_W) = v_c - \ell'$. These capacities ensure that exactly $v_c - \ell'$ votes from W^c are kept in the final election.
- $b_\beta(d) = |V \cup W|$ and $b_\alpha(d) = v_c - |(V^{\neg c} \cup W^{\neg c})^d|$ for every candidate $d \in C \setminus \{c\}$. As discussed above, all votes in $V^{\neg c} \cup W^{\neg c}$ are in the final elections. These votes give $|(V^{\neg c} \cup W^{\neg c})^d|$ vetoes to the candidate d. Hence, the lower-bound capacity for d is to ensure that in the final election d has at least the same number of vetoes as c. The upper-bound capacity for d is not important and can be changed to any integer that is larger than the maximum possible vetoes the candidate d can obtain.

It is fairly easy to check that there is a b-edge cover with at most v_c edges if and only if c can be made a winner in the final election by replacing exactly $|V^c| - \ell'$ votes.

Case 3: $\ell \leq \min(|V^c|, |W| - |W^c|$.

In this case, the optimal choice for the chair is to replace exactly ℓ voters in V vetoing c with ℓ voters from W not vetoing c. In other words, we have ensured that the final election contains all voters in $V^{\neg c}$, exactly $|V^c| - \ell$ voters in V^c, and exactly ℓ voters from $W^{\neg c}$. This observation enables us to reduce the 2-Veto-CCRV instance in this case to the following b-EC instance.

The vertex set is $\{c_V\} \cup (C \setminus \{c\})$, i.e., we create a vertex c_V first and then for each candidate in $C \setminus \{c\}$ we create a vertex denoted by the same symbol. For each voter in V^c vetoing some $d \in C \setminus \{c\}$ (and c), we create an edge (c_V, d). In addition, for each voter in $W^{\neg c}$ vetoing two distinct candidates d and e, we create an edge (d, e). The capacities of the vertices are as follows:

- $b_\alpha(c_V) = b_\beta(c_V) = |V^c| - \ell$. This capacity makes sure that exactly $|V^c| - \ell$ voters from V^c remain in the final election.
- For every $d \in C \setminus \{c\}$, we set $b_\beta(d) = |V \cup W|$ and

$$b_\alpha(d) = \max\left(0, |V^c| - \ell - |(V^{\neg c})^d|\right).$$

The lower bound ensures that in the final election d has at least the same number of vetoes as c. Here, $|(V^{\neg c})^d|$ is the number of vetoes of d obtained from voters in $V^{\neg c}$ which, as discussed above, are ensured in the final election. The upper bound is not very important and can be set as any integer larger than the maximum possible number of vetoes that d can obtain in the final election.

Given the above discussions, it is fairly easy to check that c can be made a winner by replacing ℓ voters if and only if there is a b-edge cover of size at most $|V^c|$.

Each subcase can be done in polynomial time. Consequently, the overall algorithm terminates in polynomial time. Since we thus have a polynomial-time reduction from 2-Veto-CCRV to b-EC and b-EC can be solved in polynomial time, the theorem is proven. □

We fill the complexity gap of CCRV for k-veto by showing that k-Veto-CCRV is NP-complete for every $k \geq 3$. The proof is an adaption of the NP-hardness proof of constructive control by adding voters for 3-veto due to Lin [39].[8]

Theorem 6 *For every constant $k \geq 3$, k-Veto-CCRV is* NP*-complete.*

Proof We show our result only for $k = 3$ and argue at the end of the proof how to handle the cases $k \geq 4$. Our proof provides a reduction from the RX3C problem. Given an instance $(U, \mathcal{S})$ of RX3C, where $|U| = |\mathcal{S}| = 3\kappa$, we construct an instance of 3-Veto-CCRV as follows. Let the candidate set be $C = \{c\} \cup \{d_1, d_2, d_3\} \cup U$, where the set

[8] We remark in passing that Loreggia et al. [43] showed NP-hardness for k-Approval-CCRV with $k \leq m - 3$ from which NP-hardness of k-Veto-CCRV with $k \geq 3$ immediately follows (k-veto and $(m - k)$-approval are the same for constant m), but their proof (given in the PhD thesis of Loreggia [42]), which reduces X3C to 3-Approval-CCRV, does not make it clear how the reduction can be adapted to k-approval with $k \leq m - 3$ (in particular, since the addition of dummy candidates would also increase m).

$\{c, d_1, d_2, d_3\}$ is disjoint from U. The distinguished candidate is c. For ease of exposition, let $n = 3\kappa$. The multiset V consists of the following $2n - 2\kappa + 3\kappa n$ registered voters:

- There are $n + \kappa$ voters vetoing c, d_1, and d_2;
- There are n voters vetoing d_1, d_2, and d_3; and
- For each $u \in U$, there are $n - 1$ voters vetoing u and any two arbitrary candidates in $\{d_1, d_2, d_3\}$.

Note that with the registered voters, the distinguished candidate c has $n + \kappa$ vetoes, each $u \in U$ has $n - 1$ vetoes, and d_i, $i \in \{1, 2, 3\}$, has at least n vetoes. Let the multiset W of unregistered voters consist of the following n voters. For each $S \in \mathscr{S}$, there is a voter vetoing the candidates in S. Finally, we are allowed to replace at most κ voters, i.e., $\ell = \kappa$.

We claim that c can be made a 3-veto winner by replacing at most κ voters if and only if an exact 3-set cover of U exists.

($\Leftarrow$) Assume that U has an exact 3-set cover $\mathscr{S}' \subseteq \mathscr{S}$. After replacing the κ votes corresponding to $\mathscr{S}'$ from W with κ voters in V vetoing c, c has $(n + \kappa) - \kappa = n$ vetoes, every $u \in U$ has $(n - 1) + 1 = n$ vetoes, and each d_1, d_2, and d_3 has at least n vetoes. Clearly, c becomes a winner.

($\Rightarrow$) Assume that c can be made a 3-veto winner by replacing at most ℓ voters. Let $V' \subseteq V$ and $W' \subseteq W$ be the two multisets such that $|V'| = |W'|$ and c becomes a winner after replacing all votes in V' with all votes in W'. Observe first that $|V'|$ and $|W'|$ must be exactly κ, since otherwise c has at least $n + 1$ vetoes and there exists one $u \in U$ having at most $n - 1$ vetoes in the final election, contradicting that c becomes a winner in the final election. In addition, no matter which κ voters are in W', there must be at least one candidate $u \in U$ who has at most n vetoes after the replacement. This implies that each voter in V' must veto c. As a result, c has $(n + \kappa) - \kappa = n$ vetoes after the replacement. This further implies that, for each $u \in U$, there is at least one voter in W' who vetoes u. As $|W'| = \kappa$, due to the construction of W, the collection of the 3-subsets corresponding to the κ voters in W' form an exact 3-set cover.

To show NP-hardness of k-Veto-CCRV for $k \geq 4$, we additionally create $k - 3$ dummy candidates being vetoed by every vote. The correctness argument is analogous.

Turning now to control by replacing candidates in k-veto, Loreggia et al. [43] solved the two cases of constructive and destructive control by replacing candidates for veto only (i.e., for k-veto with $k = 1$). Note that Loreggia et al. [43] solved both cases for k-approval for any k. However, this does not solve these two cases for k-veto since their proofs (which again can be found in the PhD thesis of Loreggia [42]) rely on the fact that k-approval satisfies IBC, but k-veto does not.[9] We solve these two cases, CCRC and DCRC, for k-veto with $k \geq 2$ in Theorems 7 and 8.

Theorem 7 *For every constant $k \geq 2$, k-Veto-CCRC is* NP*-complete.*

[9] Indeed, to see that k-veto does not satisfy IBC, consider the set $C = \{a, b, c_1, \ldots, c_k\}$ of candidates and let there be only one voter with vote $a\ b\ c_1\ \cdots\ c_k$. Then a and b win the election under k-veto, but if we remove the bottom ranked candidate c_k, only a wins the election alone, so the set of winners can be changed by removing a bottom-ranked candidate.

Proof To prove NP-hardness of k-Veto-CCRC for $k \geq 2$, we will modify the reduction provided by Lin [39] 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 $(U, \mathscr{S}, \kappa)$ be an instance of Hitting-Set with $U = \{u_1, \ldots, u_s\}$, $s \geq 1$, $\mathscr{S} = \{S_1, \ldots, S_t\}$, $t \geq 1$, and integer κ, $1 \leq \kappa < s$ (without loss of generality, we may assume that $\kappa < s$ since $(U, \mathscr{S}, \kappa)$ is trivially a YES-instance if $\kappa \geq s$).

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

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

and unregistered candidates U. Let V contain the following votes:

- $(t + 2s)(s - \kappa + 1)$ votes of the form $Y \cdots c\ C'$;
- $(t + 2s)(s - \kappa + 1) - s + \kappa$ votes of the form $Y \cdots d\ X$;
- for each $i, 1 \leq i \leq t$, one vote of the form $Y \cdots c\ X\ S_i$;
- for each $i, 1 \leq i \leq s$, one vote of the form $Y \cdots d\ X\ u_i$; and
- for each $i, 1 \leq i \leq s$, $(t + 2s)(s - \kappa + 1) + \kappa$ votes of the form $Y \cdots c\ U \setminus \{u_i\}\ X\ u_i$.

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

candidates in C	c	d	$c' \in C'$	$x \in X$	$y \in Y$
number of vetoes	$M(s+1) + s\kappa + t$	$M + \kappa$	M	$M(s+1) + \kappa(s+1) + t$	0

We show that $(U, \mathscr{S}, \kappa)$ is a YES-instance of Hitting-Set if and only if c can be made a k-veto winner of the election by replacing κ candidates from C with candidates from U.

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

($\Leftarrow$) Assume c can be made a k-veto winner of the election by replacing κ candidates. Since the κ 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 κ candidates from U. Note that c cannot have more than $(t + 2s)(s - \kappa + 1)$ vetoes, for otherwise c would lose to the candidates from C'. Let $U' \subseteq U$ be the set of κ candidates from U that are added to the election. Since $|U'| = \kappa > 0$, c will lose all $s((t + 2s)(s - \kappa + 1) + \kappa)$ 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 κ added candidates from U need to be a hitting set of $\mathscr{S}$. Also note that with the κ added candidates from U, c also ties d (who lost κ vetoes from the fourth group of voters) and beats the candidates from X and the added candidates from U. □

The same result can be shown for destructive control by replacing candidates in k-veto elections via a similar proof.

Theorem 8 *For every constant $k \geq 2$, k-Veto-DCRC is* NP*-complete.*

Proof As in the proof of Theorem 7, 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 [39] to show that k-Veto-DCAC and k-Veto-DCDC are NP-hard.

Let $(U, \mathscr{S}, \kappa)$ be an instance of Hitting-Set with $U = \{u_1, \ldots, u_s\}$, $s \geq 1$, $\mathscr{S} = \{S_1, \ldots, S_t\}$, $t \geq 1$, and integer $\kappa, 1 \leq \kappa \leq s$.

We construct an instance $((C \cup U, V), c, \kappa)$ of k-Veto-DCRC with candidates $C = \{c, c'\} \cup X \cup Y$, where $X = \{x_1, \ldots, x_{k-1}\}$ and $Y = \{y_1, \ldots, y_\kappa\}$, and unregistered candidates U. Let V contain the following votes:

- $2(s - \kappa) + 2t(\kappa + 1) + 4$ votes of the form $\cdots\ c\ Y\ X\ c'$;
- $2t(\kappa + 1) + 5$ votes of the form $\cdots\ c'\ X\ c$;
- for each $i, 1 \leq i \leq t$, $2(\kappa + 1)$ votes of the form $\cdots\ c'\ X\ S_i$;
- for each $i, 1 \leq i \leq s$, two votes of the from $\cdots\ c\ Y\ X\ u_i$;
- for each $i, 1 \leq i \leq \kappa$, $2(s - \kappa) + 2t(\kappa + 1) + 6$ votes of the form $c\ c'\ \cdots\ y_i\ X$; and
- for each $i, 1 \leq i \leq s$, $2(s - \kappa) + 2t(\kappa + 1) + 6$ votes of the form $c\ c'\ \cdots\ u_i\ X$.

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

To complete the proof of Theorem 8, we will now show that $(U, \mathscr{S}, \kappa)$ 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 κ candidates from C with candidates from U.

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

($\Leftarrow$) Assume c can be prevented from being a k-veto winner of the election by replacing at most κ candidates. We first argue why we must remove all κ 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 κ candidates from Y need to be removed, for otherwise c does not gain any vetoes. Then κ candidates from U need to be added to the election. Note that c will always gain $2(s - \kappa)$ vetoes from those replacements, which will bring c to $2(s - \kappa) + 2t(\kappa + 1) + 5$ 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 $S_i \in \mathscr{S}$, at least one $u_j \in S_i$ needs to be added to the election. Thus the κ added candidates from U need to correspond to a hitting set of $\mathscr{S}$.

□

Table 7 Complexity of control for plurality with runoff. All results are ours. "NPC" stands for "NP-complete" and "P" stands for "polynomial-time solvable"

CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
P	**P**	**P**	**NPC**	**NPC**	**NPC**	**P**	**P**	**P**	**NPC**	**NPC**	**NPC**

Although we do not focus on parameterized complexity [13, 51] here, we mention in passing that the proofs of Theorems 7 and 8 in fact even show W[2]-hardness of CCRC and DCRC, for k-veto with $k \geq 2$.

7 Plurality with runoff and veto with runoff

We now turn to plurality with runoff and veto with runoff, two quite common voting rules that proceed in two stages, eliminating the "weakest" candidate(s) in the first stage and then holding a runoff among the two surviving candidates for a winner to emerge. To the best of our knowledge, no results on control in plurality with runoff or veto with runoff are known to date. However, a related work has been done by Guo and Shrestha [31] who studied the complexity of control for two-stage voting rules X Then Y, where X and Y are both voting rules. Particularly, under X Then Y, the rule X is first applied and then all winning candidates under X enter a runoff election whose winners are determined by Y. Plurality (respectively, veto) with runoff can be considered as an X Then Y rule where Y is plurality (respectively, veto), and X is a rule which selects exactly two candidates with the highest plurality score (respectively, with the fewest vetoes). Nevertheless, it should be pointed out that such an X Then Y rule has not been investigated by Guo and Shrestha [31].

Our results in this section are summarized in Table 7.

We first show that the problems CCAV, CCDV, and CCRV for both plurality with runoff and veto with runoff are polynomial-time solvable when ties are broken in favor of the chair in both stages. More precisely, if several candidates are tied in the first stage, the chair has the right to select the two candidates who survive this stage, and if in the second stage $N_E(c, d) = N_E(d, c)$ for the two candidates c and d who survive the first stage, the chair is obligated to select the final winner between c and d.

Instead of showing the results separately one-by-one, we prove that a variant of the multimode control problem, τ-Exact Constructive Control by Adding and Deleting Voters, denoted by τ-ECCAV+DV, is polynomial-time solvable, where τ is either plurality with runoff or veto with runoff. In this exact variant of τ-Constructive-Multimode-Control, we require that the number of added voters and the number of deleted voters are *exactly* equal to the corresponding given integer, i.e., we require that $|V'| = \ell_{\mathrm{DV}}$ and $|W'| = \ell_{\mathrm{AV}}$. Moreover, we have $\ell_{\mathrm{AC}} = \ell_{\mathrm{DC}} = 0$ and $D = \emptyset$. Note that each of CCAV, CCDV, and CCRV is polynomial-time reducible to ECCAV+DV.

For an election (C, V), a candidate $d \in C$, and $\tau \in \{\mathrm{PRun}, \mathrm{VRun}\}$, let $\tau_{(C,V)}(d)$ be the number of voters in V approving d if τ is PRun, and be the number of voters in V vetoing d if τ is VRun. In the proof of the following theorem we will show P membership of PRun-ECCAV+DV and VRun-ECCAV+DV by reducing them to the problem Integral-Minimum-Cost-Flow (IMCF), defined in Sect. 3, which is known to be polynomial-time solvable [1].

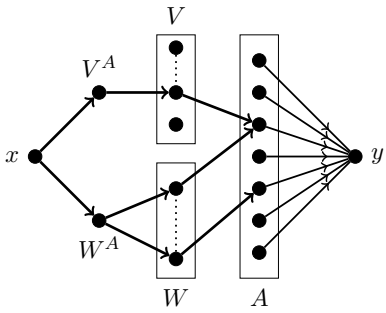

Fig. 1 An illustration of constructing the IMCF instance in the proof of Theorem 9

Theorem 9 *For each* $\tau \in \{\text{PRun}, \text{VRun}\}$, τ *-ECCAV+DV is in* P.

Proof Let (C, V), W, $c \in C$, ℓ_{AV}, and ℓ_{DV} be the components of a given instance as described in the definition of τ-ECCAV+DV. Here, c is the distinguished candidate. We first give the algorithm for τ being plurality with runoff, and then we discuss how to modify the algorithm for the case where τ is veto with runoff.

$\tau =$ **PRun.** Our algorithm tries to find a candidate $d \in C \setminus \{c\}$ and four nonnegative integers ℓ^c_{AV}, ℓ^d_{AV}, ℓ^c_{DV}, and ℓ^d_{DV} such that $\ell^c_X + \ell^d_X \leq \ell_X$ for $X \in \{\text{AV}, \text{DV}\}$. This candidate d is supposed to be the one who competes with c in the runoff stage. Moreover, ℓ^c_{AV} (respectively, ℓ^d_{AV}) is supposed to be the number of voters added from W that approve c (respectively, d), and ℓ^c_{DV} (respectively, ℓ^d_{DV}) is supposed to be the number of voters deleted from V that approve c (respectively, d). Given such a candidate and integers, we determine whether we can add exactly ℓ_{AV} votes from W of which ℓ^c_{AV} (respectively, ℓ^d_{AV}) approve c (respectively, d), and delete exactly ℓ_{DV} votes from V of which ℓ^c_{DV} (respectively, ℓ^d_{DV}) approve c (respectively, d). Clearly, the original instance is a YES-instance if and only if at least one of these tests leads to a YES answer. We show how to find the answer to each subinstance in polynomial time. First, we immediately discard a currently tested candidate d if one of the following conditions holds:

- $\ell^c_{\text{DV}} > \tau_{(C,V)}(c)$;
- $\ell^d_{\text{DV}} > \tau_{(C,V)}(d)$;
- $\ell^c_{\text{AV}} > \tau_{(C,W)}(c)$; or
- $\ell^d_{\text{AV}} > \tau_{(C,W)}(d)$.

So let us assume that none of the above conditions holds. Then the number of voters approving c and d in the final election are determined. More precisely, the number of voters approving $e \in \{c, d\}$ is $\tau_{(C,V)}(e) + \ell^e_{\text{AV}} - \ell^e_{\text{DV}}$. For notational simplicity, for each $e \in \{c, d\}$, let $\tau(e) = \tau_{(C,V)}(e) + \ell^e_{\text{AV}} - \ell^e_{\text{DV}}$. Let

$$s = \min\{\tau(c), \tau(d)\}.$$

To ensure that c and d participate in the runoff stage, each candidate $a \in C \setminus \{c, d\}$ may have at most s approvals in total. A second condition for c to be a runoff winner against d is that c is not beaten by d in their pairwise comparison. Since there are $n' = |V| + \ell_{\text{AV}} - \ell_{\text{DV}}$ voters in the final election (C, V'), d must win at most $\lfloor n'/2 \rfloor$ duels against c. Let $A = C \setminus \{c, d\}$ and $\tau_{(C,V)}(A) = \sum_{a \in A} \tau_{(C,V)}(a)$. Moreover, for $X \in \{\text{AV}, \text{DV}\}$, let $\ell^A_X = \ell_X - \ell^c_X - \ell^d_X$. As d in turn wins $\tau(d)$ comparisons against c in all votes who

approve d, if $\lfloor n'/2 \rfloor - \tau(d) < 0$, we reject the currently tested candidate d and regard the next one. Otherwise, we search for exactly

$$\underbrace{|V| - \tau_{(C,V)}(c) - \tau_{(C,V)}(d)}_{=\tau_{(C,V)}(A)} - \ell^A_{\mathrm{DV}}$$

voters in V not deleted and approving candidates in A, and exactly ℓ^A_{AV} voters added from W and approving some $a \in A$ such that the final election contains at most $\lfloor n'/2 \rfloor - \tau(d)$ voters who approve some $a \in A$ first and prefer d over c. We solve this question by reducing it to the IMCF problem.

The construction of the IMCF instance is illustrated in Figure 1. In more detail, there is a source x, a sink y, and two nodes V^A and W^A. Moreover, each voter in $V \cup W$ approving some $a \in A$ yields a node. Additionally, each $a \in A$ yields a node a. If not mentioned otherwise, each cost is equal to zero. There is an arc from x to V^A with lower-bound and upper-bound capacities

$$b_\alpha(x, V^A) = b_\beta(x, V^A) = \tau_{(C,V)}(A) - \ell^A_{\mathrm{DV}}.$$

There is another arc from x to W^A with lower-bound and upper-bound capacities

$$b_\alpha(x, W^A) = b_\beta(x, W^A) = \ell^A_{\mathrm{AV}}.$$

Each voter $v \in V$ who approves some candidate in A yields an arc (V^A, v) with upper-bound capacity 1 and lower-bound capacity 0. The cost of this arc is equal to 1 if v prefers d to c. Analogously, we define edges from W^A to vertices w corresponding to voters in W who approve some $a \in A$. There is an arc from some $v \in V \cup W$ to some $a \in A$ with upper-bound capacity 1 and lower-bound capacity 0 if and only if v approves a. Each $a \in A$ yields an arc (a, y) with upper-bound capacity s and lower-bound capacity 0.

One can check that there is a (maximum) flow with value

$$\tau_{(C,V)}(A) - \ell^A_{\mathrm{DV}} + \ell^A_{\mathrm{AV}}$$

and (minimum) cost of at most $\lfloor n'/2 \rfloor - \tau(d)$ if and only if we can find exactly $\tau_{(C,V)}(A) - \ell^A_{\mathrm{DV}}$ (remaining) voters in V approving some $a \in A$ and exactly ℓ^A_{AV} voters added from W approving some $a \in A$ such that each $a \in A$ has at most s approvals, and a weak majority of voters prefers c to d in the final election.

$\tau = \mathbf{VRun}$. Notice that in this case, $\tau_{(C,V)}(a)$ denotes the number of voters vetoing a in the election (C, V). The algorithm is similar to the above described algorithm with the following differences. First, for $X \in \{\mathrm{AV}, \mathrm{DV}\}$, ℓ^c_{X} and ℓ^d_{x} are defined analogously but with respect to vetoes of c and d, respectively. Technically, this is achieved by replacing the occurrences of the word "approve" (respectively, "approves" and "approving" and "approvals") with the word "veto" (respectively, "vetoes" and "vetoing" and "vetoes") throughout the above algorithm. Second, we replace $\lfloor n'/2 \rfloor - \tau(d)$ marked above with $\lfloor n'/2 \rfloor - \tau(c)$. Recall that in the above algorithm, we use the condition $\lfloor n'/2 \rfloor - \tau(d) < 0$ to reject a tested candidate d, as in this case a majority of voters in the final election prefer d to c. When the rule used is veto with runoff, a majority of voters in the final election prefer d to c if $\lfloor n'/2 \rfloor - \tau(c) < 0$. Finally, in the IMCF instance constructed in the above algorithm, we change the capacity of each arc from $a \in A$ to y so that the lower- bound capacity is s', where $s' = \max\{\tau(c), \tau(d)\}$, and the upper-bound capacity is $|V \cup W|$. The reason is that in veto with runoff, the two candidates with the least vetoes survive the first stage of the

election. Therefore, if the final vetoes of c and d are both at most s' with one of them being exactly s', and c and d are the two candidates surviving the first stage, it must be the case that each other candidate has at least s' vetoes in the final election. □

The exact versions of the destructive multimode control for plurality with runoff and veto with runoff are polynomial-time solvable, too.

Theorem 10 *PRun-EDCAV+DV and VRun-EDCAV+DV are in* P.

Proof To solve a PRun-EDCAV+DV or VRun-EDCAV+DV instance I with the distinguished candidate p, we solve $m-1$ instances of the constructive exact multimode problems PRun-ECCAV+DV or VRun-ECCAV+DV, respectively, each of which takes the same input as I with only the difference that the distinguished candidate is someone in $C \setminus \{p\}$, where C is the set of candidates in the input and $m = |C|$. Moreover, all the $m-1$ instances have different distinguished candidates. Clearly, I is a YES-instance of either of the two destructive problems if and only if at least one of these $m-1$ instances of the corresponding constructive problem is a YES-instance. Due to Theorem 9, each these $m-1$ instances can be solved in polynomial time. Therefore, I can be solved in polynomial time. □

Note that for each $\mathrm{Y} \in \{\mathrm{CCAV}, \mathrm{CCDV}, \mathrm{CCRV}, \mathrm{DCAV}, \mathrm{DCDV}, \mathrm{DCRV}\}$ and for each $\mathrm{X} \in \{\mathrm{PRUN}, \mathrm{VRUN}\}$, X-Y is polynomial-time Turing-reducible to its exact version. Then, given the above results, we obtain the following corollary.

Corollary 4 *For each* $\mathrm{Y} \in \{\mathrm{CCAV}, \mathrm{CCDV}, \mathrm{CCRV}, \mathrm{DCAV}, \mathrm{DCDV}, \mathrm{DCRV}\}$, *both PRun-Y and VRun-Y are in* P.

Concerning control by adding candidates, we have the following results for plurality with runoff and veto with runoff.

Theorem 11 *PRun-CCAC*, *PRun-DCAC*, *VRun-CCAC*, *and VRun-DCAC are* NP-*complete*.

Proof We prove the theorem by reductions from the RX3C problem. Let $(U, \mathscr{S})$, where $|U| = |\mathscr{S}| = 3\kappa$, be an instance of the RX3C problem. We prove the theorem for the four different problems separately.

PRun-CCAC. For each $u \in U$, we create a registered candidate, denoted by the same symbol. In addition, we create two registered candidates, q and c, with c being the distinguished candidate. Moreover, for each $S \in \mathscr{S}$, we create an unregistered candidate, denoted by the same symbol. Regarding the votes, we create $16 + 24\kappa$ votes in total defined as follows.

- First, we create nine votes with q in the first position.
- Second, we create seven votes with c in the first position.
- Third, for each $u \in U$, we create two votes with u in the first position.

The preferences over candidates other than the top-ranked one in the above $16 + 6\kappa$ votes can be set arbitrarily.

- Finally, for each $S \in \mathscr{S}$ and each $u \in S$, we create two votes of the form $S\ u\ c\ q\ \cdots$.

We complete the construction by setting $\ell = \kappa$, i.e., we are allowed to add at most κ candidates. It remains to prove the correctness of the reduction: There is an exact 3-set cover if and only if c can be made a winner by adding up to κ candidates.

($\Rightarrow$) If there is an exact 3-set cover $\mathscr{S}' \in \mathscr{S}$, we claim that $\mathscr{S}'$ is a solution of the PRun-CCAC instance constructed above. Clearly, after adding candidates in $\mathscr{S}'$, q has 9 approvals, c has 7 approvals, every $S \in \mathscr{S}'$ has 6 approvals, and every $u \in U$ has $8 - 2 = 6$ approvals. Then, according to the definition of plurality with runoff, q and c enter the runoff stage. Clearly, a majority of voters prefer c to q, and hence c becomes the unique winner after adding all candidates in $\mathscr{S}'$.

($\Leftarrow$) Consider now the opposite direction. Observe that to ensure c to survive the first stage, at least κ candidates must be added, since otherwise there were at least one candidate $u \in U$ which receives at least 8 approvals, resulting in q and u entering the runoff stage. Let $\mathscr{S}'$ be a solution of the PRun-CCAC instance. As discussed, we have $|\mathscr{S}'| = \kappa$. If $\mathscr{S}'$ is not an exact 3-set cover, again there is a candidate $u \in U$ such that u is not in any subset of $\mathscr{S}'$. According to the construction of the instance, the candidate u receives at least 8 approvals after adding the candidates in $\mathscr{S}'$, and hence survives the first stage with q. Therefore, $\mathscr{S}'$ must be an exact 3-set cover of U.

PRun-DCAC. The reduction differs from the above proof for PRun-CCAC only in that the distinguished candidate is q. The correctness relies on the observation that candidate c is the only candidate that can preclude q from winning.

VRun-CCAC. For each $u \in U$, we create a registered candidate, denoted still by u for simplicity. In addition, we create two registered candidates c and q with c being the distinguished candidate. Hence, the set of registered candidates is $C = U \cup \{c, q\}$. The unregistered candidates are created according to $\mathscr{S}$, one for each $S \in \mathscr{S}$, denoted by the same symbol for simplicity. We create a multiset V of votes as follows.

- We create one vote of the form $\mathscr{S}\ U\ c\ q$.
- For each $u \in U$, we crate $6\kappa - 3$ votes of the form $c\ q\ \mathscr{S}\ U \setminus \{u\}\ u$.
- For each $S \in \mathscr{S}$, we create $6\kappa + 5$ votes as follows:
  - $3\kappa + 1$ votes of the form $q\ U\ c\ \mathscr{S} \setminus \{S\}\ S$;
  - $3\kappa + 1$ votes of the form $c\ U\ q\ \mathscr{S} \setminus \{S\}\ S$; and
  - three votes of the form $q\ U\ \mathscr{S} \setminus \{S\}\ c\ S$.
- For each $S = \{u_x, u_y, u_z\} \in \mathscr{S}$, we further create six votes as follows:
  - two votes of the form $c\ q\ U \setminus \{u_x\}\ \mathscr{S} \setminus \{S\}\ u_x\ S$;
  - two votes of the form $c\ q\ U \setminus \{u_y\}\ \mathscr{S} \setminus \{S\}\ u_y\ S$; and
  - two votes of the form $c\ q\ U \setminus \{u_z\}\ \mathscr{S} \setminus \{S\}\ u_z\ S$.

We are allowed to add at most κ candidates, i.e., $\ell = \kappa$. Note that in the election restricted to the registered candidates,

- c has $3\kappa \cdot (3\kappa + 1) + 9\kappa$ vetoes,
- q has $3\kappa \cdot (3\kappa + 1) + 1$ vetoes, and
- every $u \in U$ has $6\kappa + 3$ vetoes.

Hence, c is not a veto with runoff winner of the election. It remains to prove the correctness of the reduction.

($\Rightarrow$) Assume that there is an exact 3-set cover $\mathscr{S}' \subseteq \mathscr{S}$ of U. After adding the candidates in $\mathscr{S}'$, candidate q has one veto, every $S \in \mathscr{S}'$ has at least $6\kappa + 11$ vetoes, every $u \in U$ has $6\kappa + 3 - 2 = 6\kappa + 1$ vetoes, and c has 6κ vetoes. Hence, q and c move on to the runoff stage. As more voters prefer c over q, c becomes the final winner.

($\Leftarrow$) Suppose that we can add a subset $\mathscr{S}' \subseteq \mathscr{S}$ of at most κ unregistered candidates to make c a winner under veto with runoff. Observe first that $\mathscr{S}'$ must contain exactly κ candidates, since otherwise c would have at least $6\kappa + 3$ vetoes, while at least one candidate in U would have at most $6\kappa + 3 - 2 = 6\kappa + 1$ vetoes. Hence, this candidate in U and q would be the two candidates going to the runoff stage. Then, from $|\mathscr{S}'| = \kappa$, it follows that c has 6κ vetoes after adding candidates in $\mathscr{S}'$. If $\mathscr{S}'$ is not an exact 3-set cover, there must be a candidate $u \in U$ occurring in at least two subsets of $\mathscr{S}'$. Then the candidate u has at most $6\kappa + 3 - 4 = 6\kappa - 1$ vetoes, leading to q and u being the two candidates competing in the runoff stage. We can conclude that $\mathscr{S}'$ is an exact 3-set cover.

VRun-DCAC. The reduction differs from the one for VRun-CCAC only in that the distinguished candidate is q. The correctness relies on the observation that candidate c is the only candidate that can preclude q from winning.

Next, we study the complexity of control by deleting candidates for plurality with runoff and veto with runoff.

Theorem 12 *PRun-CCDC*, *PRun-DCDC*, *VRun-CCDC*, *and VRun-DCDC are* NP *-complete*.

Proof Again, letting $(U, \mathscr{S})$ with $|U| = |\mathscr{S}| = 3\kappa$ be a given RX3C instance, we separately provide our four reductions from RX3C to PRun-CCDC, PRun-DCDC, VRun-CCDC, and VRun-DCDC, respectively. Let $U = \{u_1, u_2, \ldots, u_{3\kappa}\}$. Without loss of generality, assume that $\kappa \geq 4$.

PRun-CCDC. From $(U, \mathscr{S})$, we create the following instance of PRun-CCDC. Let $C = \{c, q\} \cup U \cup \mathscr{S}$ be the set of candidates and c the distinguished candidate. We create a multiset V of $9\kappa^2 + 21\kappa + 1$ votes as follows.

- We create 2κ votes of the form $q\ u_1\ u_2\ \ldots\ u_{3\kappa}\ \mathscr{S}\ c$.
- We create $\kappa + 1$ votes of the form $q\ u_{3\kappa}\ u_{3\kappa-1}\ \ldots\ u_1\ \mathscr{S}\ c$.
- For each $u \in U$, we create $3\kappa - 3$ votes of the form $u\ U \setminus \{u\}\ \mathscr{S}\ c\ q$.
- For each $S \in \mathscr{S}$, we create three votes of the form $S\ c\ C \setminus (S \cup \{c, q\})\ q$.
- For each $S = \{u_x, u_y, u_z\} \in \mathscr{S}$, we further create six votes as follows:
  - two votes of the form $S\ u_x\ C \setminus \{c, q, u_x\}\ c\ q$;
  - two votes of the form $S\ u_y\ C \setminus \{c, q, u_y\}\ c\ q$; and
  - two votes of the form $S\ u_z\ C \setminus \{c, q, u_z\}\ c\ q$.

Furthermore, let $\ell_{\mathrm{DC}} = \kappa$. It remains to prove the correctness.

($\Rightarrow$) Assume there is an exact set cover $\mathscr{S}' \subseteq \mathscr{S}$. After deleting the candidates in $\mathscr{S}'$, q has $2\kappa + \kappa + 1 = 3\kappa + 1$ approvals, c has 3κ approvals, every remaining $S \in \mathscr{S} \setminus \mathscr{S}'$ has 9 approvals, and every $u \in U$ has $3\kappa - 3 + 2 = 3\kappa - 1$ approvals. Hence, q and c go to the runoff stage, leading to c being the final winner.

($\Leftarrow$) Assume that it is possible to make c a plurality-with-runoff winner of the election by deleting a set $C' \subseteq C \setminus \{c\}$ of at most κ candidates. Note that $q \notin C'$, since otherwise there would be two candidates in U receiving at least $3\kappa - 3 + 2\kappa = 5\kappa - 3$ and

Table 8 Plurality scores of candidates in the reduction for PRun-DCDC in the proof of Theorem 12. The numbers in the equation in each row corresponding to a candidate are the plurality scores of the candidates received respectively from the four groups of votes constructed above

	plurality scores
q	$(3\kappa+4)+0+0+0=3\kappa+4$
c	$0+0+0+0=0$
$u \in U$	$0+(3\kappa-3)+0+0=3\kappa-3$
$S \in \mathscr{S}$	$0+0+9+0=9$
h_i	$0+0+0+1=1$
a_j	$0+0+0+0=0$

$3\kappa-3+\kappa+1=4\kappa-2$ approvals, preventing c from winning. Therefore, q has at least $3\kappa+1$ approvals in the final election. Furthermore, none of the candidates in U can be deleted, i.e., $U \cap C' = \emptyset$. In fact, if we delete some candidate $u \in U$, then the candidate ranked immediately after u in the $3\kappa-3$ votes created for u (in the third voter group) would receive at least $(3\kappa-3)+(3\kappa-3)=6\kappa-6$ approvals, preventing c from winning. This means that the deletion of one candidate in U invites the deletion of all candidates in U, to make c the winner. However, we are allowed to delete at most κ candidates. In summary, we have $C' \subseteq \mathscr{S}$. After deleting the candidates in C', c has $3|C'|$ approvals. Note that $|C'| = \kappa$ must hold, since otherwise at least one candidate in U would receive more approvals than candidate c, after deleting all candidates in C'; hence, this candidate and q would be the two candidates going to the runoff stage. Therefore, we know that c receives 3κ approvals after deleting all candidates in C'. If C' is not an exact 3-set cover, there must be a candidate $u \in U$ who occurs in at least two subsets of C'. Due to the construction, candidate u receives at least $3\kappa-3+2+2=3\kappa+1$ approvals, implying that q and u are the two candidates surviving the first stage, contradicting that c is the final winner after deleting all candidates in C'. Thus C' must be an exact 3-set cover.

PRun-DCDC. The candidate set is

$$C = \{c, q\} \cup U \cup \mathscr{S} \cup \{h_1, \ldots, h_{9\kappa^2+15\kappa}\} \cup A,$$

where $A = \{a_1, \ldots, a_\kappa\}$. For two positive integers x and y such that $x < y \leq 9\kappa^2$, we define

$$H[x, y] = \{h_z \mid x \leq z \leq y\}.$$

We create in total $18\kappa^2 + 36\kappa + 4$ votes classified into the following groups.

1. There are $3\kappa+4$ votes of the form $q\ C \setminus \{q\}$.
2. For each $i \in [3\kappa]$, there are $3\kappa-3$ votes of the form

   $u_i\ H[(i-1)\cdot\kappa, i\cdot\kappa]\ C \setminus (A \cup H[(i-1)\cdot\kappa, i\cdot\kappa] \cup \{u_i, c, q\})\ c\ q\ A.$

3. For each $S \in \mathscr{S}$, $S = \{u_x, u_y, u_z\}$, where $\{x, y, z\} \subseteq [3\kappa]$, there are nine votes as follows:
   - three votes of the form $S\ c\ q\ C \setminus \{S, c, q\}$;
   - two votes of the form $S\ u_x\ c\ q\ C \setminus \{S, u_x, c, q\}$;
   - two votes of the form $S\ u_y\ c\ q\ C \setminus \{S, u_y, c, q\}$; and
   - two votes of the form $S\ u_z\ c\ q\ C \setminus \{S, u_z, c, q\}$.

4. There are $9\kappa^2 + 15\kappa$ votes denoted by $v_1, \ldots, v_{9\kappa^2+15\kappa}$ such that for every $i \in [9\kappa^2 + 15\kappa]$), the vote v_i is of the form

$$h_i \; A \; c \; q \; C \setminus (\{c, q, h_i\} \cup A).$$

Let V denote the multiset of the above constructed votes. The distinguished candidate is q. Finally, we define $\ell = \kappa$, i.e., we are allowed to delete at most κ candidates from C. The time to construct the above instance is clearly bounded by a polynomial in the size of the RX3C instance.

We are left with the proof of correctness of the reduction. It is useful to first provide a summary of the plurality scores of all candidates for a better understanding of the following arguments. We refer to Table 8 for such a summary.

Due to Table 8, q survives the first stage but c does not. One can check that q is beaten by c but beats everyone else. As a consequence, q is the winner of the above constructed election.

($\Rightarrow$) Assume that there is an exact set cover $\mathscr{S}' \subseteq \mathscr{S}$ of U. Let $E = (C \setminus \mathscr{S}', V)$. We claim that q is no longer the winner of the election E. With the help of Table 8 one can check easily that in the election E the two candidates q and c receive the most approvals. Particularly, if a candidate $S \in \mathscr{S}'$ is deleted, the three votes of the form $S \; c \; q \; C \setminus \{S, c, q\}$ give three approvals to c. Then, as $|\mathscr{S}'| = \kappa$, after deleting the candidates in $\mathscr{S}'$, the candidate c receives 3κ new approvals. In addition, as $\mathscr{S}'$ is an exact set cover, for every $u \in U$, there is exactly one $S \in \mathscr{S}'$ such that $u \in S$. Then, due to the construction of the votes in the third group, the plurality score of u increases by exactly two, reaching to $3\kappa - 3 + 2 = 3\kappa - 1$. Other candidates clearly have only constant plurality scores. Therefore, c and q are the two candidates that survive the first stage, and this is the case no matter which tie-breaking scheme is used. As c beats q in the election E, we know that q is no longer a winner.

($\Leftarrow$) Assume that there is a subset $C' \subseteq C \setminus \{q\}$ of at most κ candidates such that q is no longer a winner of $(C \setminus C', V)$. First, it is easy to verify that it is impossible to prevent q from surviving the first stage by deleting at most κ candidates. Additionally, candidate c is the only one beating q. Due to these two observations, we know that the candidates surviving the first stage of $(C \setminus C', V)$ must be c and q. By Table 8, there are candidates in U who receive at least $3\kappa - 3$ approvals in E. This means that the deletion of the candidates in C' increases the plurality score of c to at least $3\kappa - 3$. Note that after deleting candidates in C', none of the votes in the groups (1), (2), and (4) rank c in the top. Therefore, the plurality score of c must be from votes in the group (3). Another significant observation is that $C' \subseteq \mathscr{S}$ and, moreover, $|C'| = \kappa$, since otherwise at least one candidate in U has a higher plurality score than that of c in E. Therefore, we know that in the election E, c has plurality score exactly 3κ. Finally, we claim that C' is an exact set cover of U. Assume for the sake of contradiction that this is not the case. Then there exists at least one candidate $u \in U$ such that there are two $S, S' \in C'$ such that $u \in S \cap S'$. By the construction of the votes in the group (3), the candidate u will be ranked in the top in four votes (two of the form $S \; u \; c \; q \; C \setminus \{S, u, c, q\}$ and two of the form $S' \; u \; c \; q \; C \setminus \{S', u, c, q\}$). This means that in the election E, the plurality score of u is at least $3\kappa - 3 + 4 = 3\kappa + 1$, which is larger than that of c. However, in this case, c is excluded in the first stage, a contradiction.

VRun-DCDC. The candidate set is the same as in the reduction for PRun-CCDC. Precisely, we define

$$C = \{c, q\} \cup U \cup \mathscr{S},$$

Table 9 Vetoes of candidates in the instance of VRun-DCDC in the proof of Theorem 12

	vetoes
q	0
c	3
$u \in U$	2
$S \in \mathscr{S}$	6

where q is the distinguished candidate. We create the following votes.

- There are three votes of the form $q \; \mathscr{S} \; U \; c$.
- For each $S = \{u_x, u_y, u_z\} \in \mathscr{S}$, we create six votes as follows:
  - two votes of the form $c \; q \; U \setminus \{u_x\} \; \mathscr{S} \setminus \{S\} \; u_x \; S$;
  - two votes of the form $c \; q \; U \setminus \{u_y\} \; \mathscr{S} \setminus \{S\} \; u_y \; S$; and
  - two votes of the form $c \; q \; U \setminus \{u_z\} \; \mathscr{S} \setminus \{S\} \; u_z \; S$.
- For each $u \in U$, there are two votes of the form $c \; q \; \mathscr{S} \; U \setminus \{u\} \; u$.

Finally, we define $\ell = \kappa$, i.e., we are allowed to delete at most κ candidates from $C \setminus \{q\}$. Clearly, the above instance of VRun-DCDC can be constructed in polynomial time. We show that there is an exact set cover of U if and only if the above VRun-DCDC instance is a YES-instance. The number of vetoes of all candidates are summarized in Table 9.

From Table 9, we know that q and some $u \in U$ survives the first stage of the election. In addition, it is easy to verify that q beats everyone else except c, and hence q wins the election.

($\Rightarrow$) Assume that U admits an exact set cover $\mathscr{S}' \subseteq \mathscr{S}$. Let $E' = (C \setminus \mathscr{S}', V)$. We claim that q is no longer a winner in the election E'. To this end, let us first analyze the vetoes of candidates in E'. Observe that deleting candidates only in $\mathscr{S}$ never changes the vetoes of c and q. So, the vetoes of q and c in E' are still 0 and 3, respectively. For each $u \in U$, as $\mathscr{S}'$ is an exact set cover of U, there is exactly one $S \in \mathscr{S}'$ such that $u \in S$. Then, after deleting S from C, u receives two more vetoes from the two votes of the form $c \; q \; U \setminus \{u\} \; \mathscr{S} \setminus \{S\} \; u \; S$, resulting in a final veto count of $2 + 2 = 4$. As this holds for all candidates in U, the two candidates surviving the first stage of the election are q and c. As pointed out above, c beats q, and hence c substitutes q as the new winner in E'.

($\Leftarrow$) Assume that there is a subset $C' \subseteq C \setminus \{q\}$ of at most κ candidates such that q is no longer a winner of $(C \setminus C', V)$ under veto with runoff. Let $E' = (C \setminus C', V)$. From Table 9, it holds that every candidate in $C \setminus C'$ except q has a positive veto count in E'. Moreover, as in each of the above constructed votes there are more than $\kappa + 1$ candidates ranked after q and $|C'| \leq \kappa$, in the election E', q has no vetoes. This means that q survives the first stage of E'. Then, as c is the only candidate that beats q, we know that c is the other candidate who survives the first stage together with q. This implies that $c \notin C'$. As in each vote not vetoing c, there are more than $\kappa + 1$ candidates ranked after c, and it holds that $|C'| \leq \kappa$, we know that the veto count of c in E' is 3. Let $\mathscr{S}' = C' \cap \mathscr{S}$ and $U' = U \setminus \bigcup_{S \in \mathscr{S}'} S$. We first prove the following claims.

Claim 1 $U' \subseteq C'$.

Assume for the sake of contradiction there exists a candidate $u \in U'$ such that $u \notin C'$. Then, due to the definition of the votes, u has two vetoes in E'. However, this contradicts with the fact that c is the candidate that survives the first stage with q. This proves Claim 1.

Claim 2 $U' = \emptyset$.

Let $t = |C' \cap \mathscr{S}|$ and $t' = |C' \cap U|$. If $U' \neq \emptyset$, then we have $t < \kappa$. As $\mathscr{S}$ covers at most $3t$ elements of U, it holds that $t' \geq 3\kappa - 3t$. It follows that $t + t' \geq 3\kappa - 2t > \kappa$, a contradiction. This proves Claim 2.

Due to the above claim, we know that $\mathscr{S}$ covers U. Then, as $|\mathscr{S}| \leq \kappa$, $\mathscr{S}$ must be an exact set cover of U.

VRun-CCDC. The reduction for VRun-CCDC is similar to the above reduction for VRun-DCDC with only the difference that we set c as the distinguished candidate. If U admits an exact set cover, then as shown above, after deleting the candidates corresponding to this set cover, c becomes the winner. For the other direction, one observes first that the above two claims still hold in this case. Then it is easy to see that if c becomes a winner after deleting at most κ candidates, the deleted candidates must correspond to an exact set cover of U.

Finally, we study the complexity of control by replacing candidates for plurality with runoff and veto with runoff.

Observe that plurality with runoff is unanimous. Then the NP-hardness result for PRun-CCAC studied in Theorem 11 and Lemma 2 directly yields NP-hardness of PRun-CCRC . In addition, plurality with runoff satisfies IBC when ties are broken deterministically. Hence, from Lemma 1 and the NP-hardness of PRun-DCDC stated in Theorem 12, it follows that PRun-DCRC is NP-hard when ties are broken deterministically. However, in the proof of NP-hardness of PRun-DCDC, the distinguished candidate q has a strictly higher plurality score than any other candidate. So, no matter which tie-breaking scheme is used, q survives the first stage. In addition, as c is the candidate who replaces q as the winner in the final election, it does not matter which candidate in U survives the first stage with q in the original election. Therefore, NP-hardness applies to all tie-breaking schemes. (Precisely, we modify the instance of PRun-DCDC by adding an additional set of κ unregistered candidates who are ranked after all the other candidates in all votes.)

However, it is easy to check that veto with runoff is not unanimous and does not satisfy IBC either. Hence, we cannot obtain NP-hardness for VRun-CCRC and VRun-DCRC using Lemmas 1 and 2. Nevertheless, we can show NP-hardness of these problems by modifications of the proofs for VRun-CCAC and VRun-DCDC studied in Theorems 11 and 12. In particular, to obtain NP-hardness of VRun-CCRC, we modify the instance of VRun-CCAC by adding an additional set of κ registered candidates and rank them before all the other candidates in all votes. More importantly, we rank all the κ registered candidates in an arbitrary but fixed order so that they have to be replaced to guarantee the victory of the distinguished candidate. To obtain NP-hardness of VRun-DCRC, we modify the instance of VRun-DCDC by creating a set of κ unregistered candidates, and rank them directly after q in all votes (i.e., q and these κ candidates are ranked consecutively in all votes with q being the first one among them). The relative order among these κ candidates does not matter.

Summing up, we have the following results.

Theorem 13 *PRun-CCRC, PRun-DCRC, VRun-CCRC, and VRun-DCRC are* NP*-complete.*

Table 10 Complexity of control for Condorcet. Our results are in boldface. "NPC" stands for "NP-complete," "P" for "polynomial-time solvable," and "I" for "immune"

CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
NPC	*NPC*	**NPC**	*I*	*P*	**P**	*P*	*P*	**P**	*P*	*I*	**P**

Note that the NP-hardness results in the above three theorems (Theorems 11, 12, and 13) hold regardless of the tie-breaking rule used because no tie occurs in either stage of the constructed elections.

8 Condorcet voting

In this section, we study Condorcet voting. Our results of this section are summarized in Table 10.

For Condorcet we will show that it is vulnerable to three types of replacement control, yet resistant to the fourth one, starting with the resistance proof.

Theorem 14 *Condorcet-CCRV is* NP*-complete.*

Proof We prove NP-hardness by reducing RX3C to Condorcet-CCRV.[10] Let $(U, \mathscr{S})$ be an RX3C instance with $U = \{u_1, \ldots, u_{3\kappa}\}$, $\kappa \geq 2$ (which may be assumed, as RX3C is trivially solvable when $\kappa = 1$), and $\mathscr{S} = \{S_1, \ldots, S_{3\kappa}\}$. The set of candidates is $C = U \cup \{c\}$ with c being the distinguished candidate. The votes are constructed as follows:

- There are $2\kappa - 3$ registered votes of the form $u_1 \ \cdots \ u_{3\kappa}\ c$ in V and
- for each $j, 1 \leq j \leq 3\kappa$, there is one unregistered vote of the form $S_j\ c\ U \setminus S_j$ in W.

The ordering of candidates in S_j and $U \setminus S_j$ does not matter in any of those votes. Finally, set $\ell = \kappa$.

Analyzing the election (C, V), u_1 is the Condorcet winner; in particular, c loses against every $u_i \in U$ with a deficit of $2\kappa - 3$ votes, i.e.,

$$N_{(C,V)}(u_i, c) - N_{(C,V)}(c, u_i) = 2\kappa - 3.$$

We will now show that $(U, \mathscr{S})$ is a YES-instance of RX3C if and only if c can be made the Condorcet winner of the election by replacing κ votes from V with votes from W.

($\Rightarrow$) Assume there is an exact cover $\mathscr{S}' \subseteq \mathscr{S}$ of U. We remove κ votes of the form $u_1 \ \cdots \ u_{3\kappa}\ c$ from the election and replace them by the votes of the form $S_j\ c\ U \setminus S_j$ for all $S_j \in \mathscr{S}'$. Let (C, V') be the resulting election. Since $\mathscr{S}'$ is an exact cover of U, for each $u_i \in U$,

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

[10] A similar reduction was used by Bartholdi, Tovey, and Trick [7] to prove that Condorcet-CCAV is NP -hard.

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

($\Leftarrow$) Assume that c can be made a Condorcet winner of the election by replacing at most κ votes. Recall that c has a deficit of

$$N_{(C,V)}(u_i, c) - N_{(C,V)}(c, u_i) = 2\kappa - 3$$

to every $u_i \in U$ in the original election. Thus *exactly* κ votes need to be removed from the election, for otherwise c's deficit of at least $\kappa - 2$ 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 κ removed votes, c's deficit to every other candidate is now decreased to $\kappa - 3$. However, none of the κ votes from W replacing the removed votes can rank some $u_i \in U$ in front of c more than once, as otherwise we would have

$$N_{(C,V')}(u_i, c) \geq \kappa - 1 > \kappa - 2 \geq N_{(C,V')}(c, u_i)$$

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

By contrast, we show vulnerability to destructive control by replacing voters for Condorcet via a simple algorithm.

Theorem 15 *Condorcet-DCRV is in* P.

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 $(C, V \cup W)$, the distinguished candidate $c \in C$, and a positive integer ℓ. The algorithm will output either a pair (V', W') with $V' \subseteq V$, $W' \subseteq W$, and $|V'| = |W'| \leq \ell$ (i.e., for c to not win, there are votes in V' that must be removed and votes in W' that must be added to the election instead), 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 $N_{(C,V)}(c, d) - N_{(C,V)}(d, c) + 1 \leq 2\ell$ (if this is not the case, d loses to c in any case and we can skip this candidate.)

Let $V' \subseteq V$ contain at most ℓ votes from V preferring c to d and let $W' \subseteq W$ contain at most ℓ votes from W 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 $N_E(c, d) \leq N_E(d, c)$ in the election $E = (C, (V \cup W') \setminus V'))$. If this is the case, c does not beat d in direct comparison, so c cannot win the election. The algorithm then outputs (V', W').

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 [7] 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 [33] 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, as shown in the following example.

Example 1 To see that Condorcet is susceptible to constructive control by replacing candidates, consider a set $C = \{b, c\}$ with two registered candidates, a set $D = \{d\}$ with just one unregistered candidate, and only one vote of the form $b\ c\ d$ 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 16 *Condorcet-CCRC is in* P.

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 unregistered candidates for c to win.

The input to our algorithm is an election $(C \cup D, V)$, the distinguished candidate $c \in C$, and a positive integer ℓ. The algorithm will output either a pair (C', D') with $C' \subseteq C \setminus \{c\}$, $D' \subseteq D$, and $|C'| = |D'| \leq \ell$ (i.e., for c to win, there are candidates in C' that must be removed and candidates in D' that must be added to the election instead), 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 $C' \subseteq C \setminus \{c\}$ be the set of candidates from $C \setminus \{c\}$ that beat or tie c in direct comparison and let $D' \subseteq D$ be a set of at most $|C'|$ candidates from D that c beats in direct comparison.

If $|C'| \leq \ell$ and $|C'| = |D'|$, we output (C', D'), and otherwise we output "control impossible."

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

Theorem 17 *Condorcet-DCRC is in* P.

Proof An algorithm that solves the problem works as follows: Given an election $(C \cup D, V)$, a distinguished candidate $c \in C$, and an integer ℓ, it checks 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 $d \in D$ who beats or ties c in direct comparison, whether there is another candidate $b \in C$ with $b \neq c$ and whether $\ell \geq 1$. If these conditions are satisfied, it outputs $(\{b\}, \{d\})$, and otherwise "control impossible."

Table 11 Complexity of control for fallback voting. Our results are in boldface. "NPC" stands for "NP -complete" and "P" stands for "polynomial-time solvable"

CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
NPC	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**	*P*	*P*	**P**	*NPC*	*NPC*	**NPC**

This algorithm outputs either a successful pair (C', D') with $C' \subseteq C \setminus \{c\}$, $D' \in D$, and $|C'| = |D'| \leq \ell$ if c can be prevented from winning by replacing at most ℓ candidates, or else "control impossible." Obviously, the algorithm is correct and runs in polynomial time.

9 Fallback voting

We will now consider fallback voting and show that it is vulnerable to one type of replacement control and resistant to the others. Our results for fallback voting are summarized in Table 11.

Theorem 18 *Fallback-CCRV is* NP*-complete.*

Proof To prove NP-hardness, we will modify the reduction from X3C that Erdélyi and Rothe [22] (and Erdélyi et al. [16]) used to show NP-hardness of Fallback-CCAV. Let $(U, \mathscr{S})$ be an X3C instance with $U = \{u_1, \ldots, u_{3\kappa}\}$, $\kappa \geq 2$, and $\mathscr{S} = \{S_1, \ldots, S_t\}$, $t \geq 1$. The set of candidates is $C = U \cup B \cup \{c\}$ with c being the distinguished candidate and $B = \{b_1, \ldots, b_{t(3\kappa-4)}\}$ a set of $t(3\kappa - 4)$ dummy candidates. In V (corresponding to the registered voters), there are the $3\kappa - 1$ votes (recall the input format in fallback elections described in Sect. 3):

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

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

– For each j, $1 \leq j \leq t$, let $B_j = \{b_{(j-1)(3\kappa-4)+1}, \ldots, b_{j(3\kappa-4)}\}$ and include in W the vote $B_j\, S_j\, c \mid (U \setminus S_j) \cup (B \setminus B_j)$.

Finally, set $\ell = \kappa$.

Having no approvals in (C, V), c does not win. We will show that $(U, \mathscr{S})$ 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 κ votes from V with as many votes from W.

($\Rightarrow$) Suppose there is an exact cover $\mathscr{S}' \subseteq \mathscr{S}$ of U. Remove κ votes $U \mid B \cup \{c\}$ from the election and add, for each $S_j \in \mathscr{S}'$, the vote $B_j\, S_j\, c \mid (U \setminus S_j) \cup (B \setminus B_j)$ instead. Let $(C, \widehat{V})$ be the resulting election. It follows that

- $score_{(C,\widehat{V})}(b_i) \leq 2$ for every $b_i \in B$,
- $score_{(C,\widehat{V})}(u_i) = \kappa$ for every $u_i \in U$ ($\kappa - 1$ approvals from the remaining registered voters and one approval from the added voters since $\mathscr{S}'$ is an exact cover of U), and

- $score_{(C,\hat{V})}(c) = \kappa$.

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

($\Leftarrow$) Suppose c can be made a fallback winner of the election by replacing at most κ votes from V with as many votes from W. Since c has no approvals in (C, V) and we can only add at most κ 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 $\kappa \geq 2$ unregistered votes to the election. We need to remove κ votes $U \mid B \cup \{c\}$ from the election; otherwise, some $u_i \in U$ would have at least s approvals, whereas c could gain no more than $\kappa - 1$ approvals from adding unregistered votes. Each $u_i \in U$ receives $\kappa - 1$ approvals from the remaining registered votes of the original election and c receives κ approvals from the added votes. Additionally, every added voter approves of three candidates from U. Hence, in order for c to at least tie every candidate from U, each $u_i \in U$ can only be approved by at most one of the added votes. Since there are κ added votes, there must be an exact cover of U.

By contrast, we establish vulnerability of the destructive case of control by replacing voters for fallback voting. The proof employs a rather involved polynomial-time algorithm solving this problem.

Theorem 19 *Fallback-DCRV is in* P.

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 prevent the distinguished candidate from being a fallback winner. The algorithm is inspired by an algorithm designed by Erdélyi and Rothe [22] (see also Erdélyi et al. [16]) 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 $(C, V \cup W)$, a distinguished candidate $c \in C$, and an integer ℓ. The algorithm will output either a pair (V', W') with $V' \subseteq V$, $W' \subseteq W$, and $|V'| = |W'| \leq \ell$ (i.e., for c to not win, there are votes in V' that must be removed and votes in W' that must be added to the election instead), or that control is impossible.

The algorithm runs through $n = \max_{v \in V \cup W} |S_v|$ 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 ℓ 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 i, $1 \leq i \leq n$: For $i > 1$, this stage is reached if we could not successfully control the election in majority stages 1 through $i - 1$. Note that in each majority stage i

we assume that a candidate that is approved by a voter on level $j > i$ is disapproved by this voter. 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 conditions hold:

$$def^{i}_{(C,V)}(d) \leq \ell; \tag{1}$$

$$score^{i}_{(C,V)}(d) > score^{i}_{(C,V)}(c) - 2\ell. \tag{2}$$

If at least one of (1) and (2) does not hold, d cannot have a strict majority on level i or cannot beat c on this level, no matter which at most ℓ 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 (i.e., if both (1) and (2) hold), we determine the largest $W_d \subseteq W$ such that $|W_d| \leq \ell$ and all votes of W_d approve of d and disapprove of c on the first i levels. Furthermore, we determine the largest $V_d \subseteq V$ such that $|V_d| \leq \ell$ and all votes of V_d approve of c and disapprove of d on the first i levels. Again, if $|V_d| \neq |W_d|$, we fill up the smaller vote list with votes as follows until they are equal in size:

- If $|V_d| < |W_d|$, we fill up V_d with votes of $V \setminus V_d$ who approve of neither c nor d until we either have $|V_d| = |W_d|$ 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 W_d until both lists are equally large.
- If $|V_d| > |W_d|$, we fill up W_d with votes of $W \setminus W_d$ 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 $W \setminus W_d$ to W_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) until both lists are equally large.

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

$$score^{i}_{(C,(V\setminus V_d)\cup W_d)}(d) \geq maj(V); \tag{3}$$

$$score^{i}_{(C,(V\setminus V_d)\cup W_d)}(d) > score^{i}_{(C,(V\setminus V_d)\cup W_d)}(c). \tag{4}$$

If (3) or (4) does not hold, 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:

$$score^{i-1}_{(C,(V\setminus V_d)\cup W_d)}(c) \geq maj(V). \tag{5}$$

If (5) does not hold, we output (V_d, W_d), as d wins on the ith level and so prevents c from winning. Note that for $i = 1$ condition (5) always fails to hold, so the following steps are only done in majority stages 2 through n. If (5) 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 W_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 W_d and one from V_d that approve of c on level i as long as possible and as long as

$$score^{i}_{(C,(V\setminus V_d)\cup W_d)}(d) \geq maj(V)$$

and (4) still hold. Note that we can skip this step if W_d was not filled up with votes in earlier steps to bring W_d and V_d to the same size.

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

(a) $|V_d \cup V_{cd}| \leq \ell$,
(b) $|V_{cd}| = |W_{cd}|$,
(c) all votes in V_{cd} approve of c on the first $i-1$ levels,
(d) all votes in W_{cd} approve of c on level i or disapprove of c, and
(e) we have

$$score^{i}_{(C,(V\setminus(V_d\cup V_{cd}))\cup W_d\cup W_{cd})}(d) \geq \max\{maj(V), score^{i}_{(C,(V\setminus(V_d\cup V_{cd}))\cup W_d\cup W_{cd})}(c)+1\}.$$

Items (a), (b), and (e) make sure that we still have a valid replacement action and items (c) and (d) find the best votes to be added and removed such that c loses approvals on the first $i-1$ levels.

Then we check the following condition:

$$score^{i-1}_{(C,(V\setminus(V_d\cup V_{cd}))\cup W_d\cup W_{cd})}(c) \geq maj(V). \tag{6}$$

If (6) holds, c cannot be prevented from reaching a strict majority in the first $i-1$ levels without d not reaching a strict majority or failing to beat c on level i 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 $(V_d \cup V_{cd}, W_d \cup W_{cd})$ as a successful pair.

ApprovalStage: This stage will only be reached if it was not possible to find a successful control action in majority stages 1 through n.

We first check whether the following holds:

$$score_{(C,V)}(c) - \ell < maj(V). \tag{7}$$

If (7) does not hold, we output "control impossible" since, after replacing at most ℓ 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 ℓ votes are replaced.

Otherwise (i.e., if (7) holds), we iterate over all candidates $d \in C \setminus \{c\}$ and check whether

$$score_{(C,V)}(c) - 2\ell > score_{(C,V)}(d).$$

If this is 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 his or 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 $W_d \subseteq W$ such that $|W_d| \leq \ell$ and all votes of W_d approve of d and disapprove of c. Furthermore, we again determine the largest $V_d \subseteq V$ such that

$|V_d| \leq \ell$ and all votes of V_d approve of c and disapprove of d. Once more, if $|V_d| \neq |W_d|$, we fill up the smaller vote list with votes as follows until they are equal in size:

- If $|V_d| < |W_d|$, we fill up V_d with votes of $V \setminus V_d$ who approve of both c and d until we either have $|V_d| = |W_d|$ 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 W_d until both lists are equally large.
- If $|V_d| > |W_d|$, we fill up W_d with votes of $W \setminus W_d$ that disapprove of both c and d until we either have $|V_d| = |W_d|$ or run out of those votes, and in the latter case we now keep adding to W_d those votes of $W \setminus W_d$ that approve of both c and d. We prefer adding votes disapproving both c and d over votes approving both c and d since the former type of votes keep c's score as low as possible. Again, only if this is still not enough to make both vote lists equal in size, we remove votes from V_d until both lists are equally large. Afterwards, if there are votes in $V \setminus V_d$ that approve of both c and d and votes in $W \setminus W_d$ that disapprove of both c and d, we add as many as possible of them to V_d and W_d, respectively, always ensuring that $|V_d| = |W_d|$ still holds. By doing this, we further reduce c's score without changing the score balance of c and d.

Then we check the following conditions:

$$score_{(C,(V\setminus V_d)\cup W_d)}(d) > score_{(C,(V\setminus V_d)\cup W_d)}(c), \tag{8}$$

$$score_{(C,(V\setminus V_d)\cup W_d)}(c) < maj(V). \tag{9}$$

If (8) and (9) are true, output (V_d, W_d) since we have successfully prevented c from reaching a strict majority and found a candidate d that beats c by approval score.

Otherwise, we proceed 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 20 *Fallback-CCRC and Fallback-DCRC are* NP*-complete.*

Proof Erdélyi and Rothe [22] (see also the subsequent journal version by Erdélyi et al. [16]) showed that fallback is resistant to constructive and destructive control by deleting candidates. Recall that 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 ℓ, and we ask whether c can be made a fallback winner by deleting at most ℓ votes, whereas 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 ℓ votes. To prove the theorem, we will reduce

Table 12 Complexity of control for range voting (second row) and for normalized range voting (the third row). Our results are in boldface. "NPC" stands for "NP-complete," "P" for "polynomial-time solvable," and "I" for "immune"

CCAV	CCDV	CCRV	CCAC	CCDC	CCRC	DCAV	DCDV	DCRV	DCAC	DCDC	DCRC
NPC	*NPC*	**NPC**	*I*	*P*	**P**	*P*	*P*	**P**	*P*	*I*	**P**
NPC	*NPC*	**NPC**	*NPC*	*NPC*	**NPC**	*P*	*P*	**P**	*NPC*	*NPC*	**NPC**

– Fallback-CCDC to Fallback-CCRC and
– Fallback-DCDC to Fallback-DCRC, respectively.

Let $((C, V), c, \ell)$ be an instance of Fallback-CCDC (or Fallback-DCDC). We construct from (C, V) a fallback election $(C \cup D, V')$ with (dummy) unregistered candidates $D = \{d_1, \ldots, d_\ell\}$, $D \cap C = \emptyset$, 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 the deletion bound ℓ now becomes 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 ℓ candidates from C if and only if c can be made a fallback winner of $(C \cup D, V')$ by deleting up to ℓ candidates from C and adding the same number of dummy unregistered candidates from D. This gives the desired reduction in both the constructive and the destructive case.

10 Range voting and normalized range voting

Now we study range voting and normalized range voting. Our results in this section are summarized in Table 12.

We first solve the cases in which range voting and normalized range voting have the same complexity and can be solved at one go starting with constructive control by replacing voters that follows from a result by Menton [48] that makes use of the fact that approval voting is a special case of range voting and normalized range voting.

Theorem 21 (Menton [48]**)** *If approval voting is resistant to a case of control, range voting and normalized range voting will also be resistant for any scoring range.*

Corollary 5 *Range-Voting-CCRV and Normalized-Range-Voting-CCRV are* NP*-complete.*

The destructive variant can be solved by a simple algorithm for range voting and normalized range voting.

Theorem 22 *Range-Voting-DCRV and Normalized-Range-Voting-DCRV are in* P.

Proof To prove membership in P of both problems, we will provide an algorithm that solves the problems 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 a k-range election $(C, V \cup W)$, the distinguished candidate $c \in C$, and

an integer ℓ. The algorithm will output either a pair (V', W') with $V' \subseteq V$, $W' \subseteq W$, and $|V'| = |W'| \leq \ell$ (i.e., for c to not win, there are votes in V' that must be removed and votes in W' that must be added to the election instead), 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 (i.e., if c is initially winning), we will try to find a candidate $d \in C \setminus \{c\}$ who can beat the distinguished candidate c if voters are replaced. Since removing voters from or adding voters to the election does not affect the number of points (normalized or not) other voters give to the candidates, we can compute the change of the points balance (for range voting and normalized range voting, respectively) of c and d for each voter in $V \cup W$. Formally, let $v \in V \cup W$ and s_c^v and s_d^v be the (normalized) points given to c and d by voter v. Let $dist_{(C,\{v\})}(c, d) = s_c^v - s_d^v$ be the points difference that c and d would gain if v were part of the election. Order the voters in V and W, respectively, according to those values. Let $V' = \emptyset$ and $W' = \emptyset$. Then, in at most ℓ rounds, choose one vote $v \in V$ to remove from the election that maximizes the points balance in favor of c (i.e., $v = \arg\max_{v \in V} dist_{(C,\{v\})}(c, d)$) and one vote from $w \in W$ to add to the election that maximizes the points balance in favor of d (i.e., $w = \arg\min_{v \in V} dist_{(C,\{v\})}(c, d)$). If the replacement of v with w changes the points balance of c and d in favor of d (i.e., if $dist_{(C,\{w\})}(c, d) - dist_{(C,\{v\})}(c, d) < 0$), set $V = V \setminus \{v\}$, $V' = V' \cup \{v\}$, $W = W \setminus \{w\}$, and $W' = W' \cup \{w\}$.

Afterwards, check whether c is beaten by d in $(C, (V \setminus V') \cup W')$ and output (V', W') if this is the case. If there is no such candidate d, output that control is impossible. The algorithm solves the problems and runs in polynomial-time.

Turning now to control by replacing candidates, we start by examining constructive and destructive control for range voting and show that these problems are easy to solve. First note that Menton [48] showed that range voting (just like its special variant approval voting [33]) is immune to constructive control by adding candidates and to destructive control by deleting candidates. For control by *replacing* candidates, however, range voting is susceptible both in the constructive and in the destructive case, as shown in the following example.

Example 2 Consider a set $C = \{c, d\}$ of registered candidates, a set $D = \{e\}$ with only one unregistered candidate, and one voter v with points vector (1, 2, 0), where $C \cup D$ is ordered lexicographically (i.e., c gets one point, d two, and e zero points). If we are allowed to replace one candidate, c loses in the 2-range election (C, V) under range voting, but wins if d is replaced by e. This shows that range voting is susceptible to constructive control by replacing candidates.

We can use the same candidate sets C and D and the points vector (1, 0, 2) for v to show susceptibility of range voting for destructive control by replacing candidates analogously.

Theorem 23 *Range-Voting-CCRC and Range-Voting-DCRC are in* P.

Proof For range voting, adding or removing candidates does not affect the points given to other candidates. Therefore, for an input of Range-Voting-CCRC and Range-Voting-DCRC, respectively, we do the following after checking whether the chair's constructive or destructive goal is reached trivially (and accepting in this case).

In the constructive case, we need to check whether the number of registered candidates that beat the distinguished candidate c is at most ℓ and whether there are enough

unregistered candidates that do not beat c so that each of them can replace one registered candidate beating c. If so, we accept; otherwise, control is impossible.

In the destructive case, we check if there exists an unregistered candidate d that beats c; if so, we choose an arbitrary registered candidate that is not c and replace this candidate by d; otherwise, control is impossible.

In contrast to range voting, we now show that normalized range voting is resistant to constructive and destructive control by replacing candidates. Starting with constructive control, we adapt a reduction by Menton [48] to reduce Hitting-Set to Normalized-Range-Voting-CCRC.

Theorem 24 *Normalized-Range-Voting-CCRC is* NP*-complete.*

Proof The reduction is a simple modification of the reduction that Menton [48] used to show that normalized range voting is resistant to constructive control by adding candidates.

Given a Hitting-Set instance $(U, \mathscr{S}, \kappa)$, construct a Normalized-Range-Voting-CCRC instance as follows. Let $C = E \cup \{c, w\}$ with $E = \{e_1, \ldots, e_\kappa\}$ be the set of registered candidates and $D = U$ the set of unregistered candidates.

- $2t(\kappa + 1) + 4s$ voters give a score of 2 to c and each $e_i \in E$, and a score of 0 to all other candidates;
- $3t(\kappa + 1) + 2\kappa$ voters give a score of 2 to w and each $e_i \in E$, and a score of 0 to all other candidates;
- for each $b \in U$, 4 voters give a score of 2 to b and each $e_i \in E$, a score of 1 to w, and a score of 0 to all other candidates; and
- for each $S_i \in \mathscr{S}$, $2(\kappa + 1)$ voters give a score of 2 to each $b \in S_i$ and each $e_i \in E$, a score of 1 to c, and a score of 0 to all other candidates.

The voters are exactly the same as in the reduction for Normalized-Range-Voting-CCAC of Menton [48] (the number of voters in the second group are adjusted to the nonunique-winner model) except that every voter gives the candidates from E the maximum number of points. Since w gains zero points from the second group of voters in order for w to have a chance of winning, all candidates from E need to be removed. Together with the fact that we can pad every solution of the Hitting-Set instance to contain exactly κ elements of U we can conclude that $(U, \mathscr{S}, \kappa)$ is a YES-instance of Hitting-Set if and only if $((C \cup D, V), w, \kappa)$ is a YES-instance of Normalized-Range-Voting-CCRC.

For the destructive variant we can use the NP-hardness of Normalized-Range-Voting-DCDC proven by Menton [48].

Theorem 25 *Normalized-Range-Voting-DCRC is* NP*-complete.*

Proof To show NP-hardness we will reduce Normalized-Range-Voting-DCDC to Normalized-Range-Voting-DCRC. Given a Normalized-Range-Voting-DCDC instance $((C, V), c, \ell)$, construct a set of unregistered candidates D with $|D| = \ell$ and let every voter $v \in V$ give every candidate from D as many points as he or she gives to c. Therefore, c and every candidate from D will always have the same number of points. Since c is always part of the election (removing c would trivially achieve the destructive goal), adding any candidate of D never affects the number of points given to other candidates. Therefore, if at

most ℓ candidates from $C \setminus \{c\}$ can be removed from the election (C, V) to make c not win (i.e., $((C, V), c, \ell)$ is a YES-instance of Normalized-Range-Voting-DCDC), we can add the same number of candidates from D to the election without changing the winners, so $((C \cup D, V), c, \ell)$ is a YES-instance of Normalized-Range-Voting-DCRC. For the converse direction, if we cannot make c be beaten in (C, V) by removing at most ℓ candidates, we cannot do so by adding candidates from D. Menton [48] showed that Normalized-Range-Voting-DCDC is NP-hard. Thus the theorem is proven.

11 Conclusions and open problems

We have investigated the computational complexity of control for Copeland$^{\alpha}$, maximin, k-veto, plurality with runoff, veto with runoff, Condorcet, fallback, range voting, and normalized range voting, closing a number of gaps in the literature. Table 1 on page 5 in Sect. 2 gives an overview of our and previously known results on the complexity of control by replacing, adding, and deleting either candidates or voters for the voting rules mentioned above.

Our proofs are based on the nonunique-winner model but can be modified to work for the unique-winner model of the control problems as well. Notice that the complexity of CCRV for 2-approval remains the only open problem in Table 1. The polynomial-time algorithm for 2-Veto-CCRV from the proof of Theorem 5 cannot be trivially extended to 2-approval. In 2-veto, any optimal solution only replaces registered voters in V that veto the distinguished candidate. However, this is not the case in 2-approval. In a worst case, we need to replace registered votes in V that do not approve of c with some unregistered votes in W that also do not approve of c. It is not clear how to reduce such a worst-case instance to an equivalent b-EC instance.

We point out that the complexity of partitioning either candidates or voters (in the various scenarios due to Bartholdi, Tovey, and Trick [7] and Hemaspaandra, Hemaspaandra, and Rothe [33]) is still open for plurality with runoff and veto with runoff. In addition, it would also be interesting to study the *parameterized* complexity of control problems for plurality with runoff and veto with runoff. Third, it is important to point out that our NP -completeness results provide purely a worst-case analysis and whether these problems are hard to solve in practice needs to be further investigated. Finally, our polynomial-time algorithm in Theorem 9 relies on that ties are broken in favor of the chair. It would be interesting to see if the result still holds for other tie-breaking rules. It has been observed that tie-breaking rules may affect the complexity of strategic voting problems [3, 52, 63].

Acknowledgements We thank the anonymous JAAMAS, AAMAS'19, CSR'20, and ISAIM'20 reviewers for their helpful comments. This work was supported in part by DFG Grants RO-1202/14-2 and RO-1202/21-1.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Ahuja, R., Magnanti, T., & Orlin, J. (1993). *Network Flows: Theory, Algorithms, and Applications*. New Jersey: Prentice-Hall.
2. Arora, S., & Barak, B. (2009). *Computational Complexity: A Modern Approach*. Cambridge: Cambridge University Press.
3. Aziz, H., Gaspers, S., Mattei, N., Narodytska, N., & Walsh, T. (2013) Ties matter: Complexity of manipulation when tie-breaking with a random vote. In: Proceedings of the 27th AAAI Conference on Artificial Intelligence, pp. 74–80
4. Bang-Jensen, J., & Gutin, G. (2008). *Digraphs: Theory*. Berlin: Springer-Verlag.
5. Bartholdi, J., & III., & Orlin, J. . (1991). Single transferable vote resists strategic voting. *Social Choice and Welfare, 8*(4), 341–354.
6. Bartholdi, J., & III., Tovey, C., & Trick, M. . (1989). The computational difficulty of manipulating an election. *Social Choice and Welfare, 6*(3), 227–241.
7. Bartholdi, J., & III., Tovey, C., & Trick, M. . (1992). How hard is it to control an election? *Mathematical and Computer Modelling, 16*(8/9), 27–40.
8. Baumeister, D., & Rothe, J. (2015). Preference aggregation by voting. In: J. Rothe (ed.) Economics and Computation. An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division, Springer Texts in Business and Economics, chap. 4, pp. 197–325. Springer-Verlag
9. Betzler, N., & Uhlmann, J. (2009). Parameterized complexity of candidate control in elections and related digraph problems. *Theoretical Computer Science, 410*(52), 5425–5442.
10. Brams, S., & Sanver, R. (2009). Voting systems that combine approval and preference. In: S. Brams, W. Gehrlein, F. Roberts (eds.) The Mathematics of Preference, Choice, and Order: Essays in Honor of Peter C. Fishburn, pp. 215–237. Springer
11. Chen, J., Faliszewski, P., Niedermeier, R., & Talmon, N. (2017). Elections with few voters: Candidate control can be easy. *Journal of Artificial Intelligence Research, 60,* 937–1002.
12. Conitzer, V., & Walsh, T. (2016). Barriers to manipulation in voting. In: F. Brandt, V. Conitzer, U. Endriss, J. Lang, A. Procaccia (eds.) Handbook of Computational Social Choice, chap. 6, pp. 127–145. Cambridge University Press
13. Downey, R., & Fellows, M. (2013). *Parameterized Complexity* (2nd ed.). Springer-Verlag.
14. Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001). Rank aggregation methods for the web. In: Proceedings of the 10th International World Wide Web Conference, pp. 613–622
15. Ephrati, E., & Rosenschein, J. (1997). A heuristic technique for multi-agent planning. *Annals of Mathematics and Artificial Intelligence, 20*(1–4), 13–67.
16. Erdélyi, G., Fellows, M., Rothe, J., & Schend, L. (2015). Control complexity in Bucklin and fallback voting: A theoretical analysis. *Journal of Computer and System Sciences, 81*(4), 632–660.
17. Erdélyi, G., Fellows, M., Rothe, J., & Schend, L. (2015). Control complexity in Bucklin and fallback voting: An experimental analysis. *Journal of Computer and System Sciences, 81*(4), 661–670.
18. Erdélyi, G., Hemaspaandra, E., & Hemaspaandra, L. (2015). More natural models of electoral control by partition. In: Proceedings of the 4th International Conference on Algorithmic Decision Theory, pp. 396–413
19. Erdélyi, G., Nowak, M., & Rothe, J. (2009). Sincere-strategy preference-based approval voting fully resists constructive control and broadly resists destructive control. *Mathematical Logic Quarterly, 55*(4), 425–443.
20. Erdélyi, G., Piras, L., & 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, pp. 837–844
21. Erdélyi, G., Reger, C., & Yang, Y. (2019). Towards completing the puzzle: Solving open problems for control in elections. In: Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems, pp. 846–854
22. Erdélyi, G., & Rothe, J. (2010). Control complexity in fallback voting. In: Proceedings of Computing: the 16th Australasian Theory Symposium, pp. 39–48
23. Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2011). Multimode control attacks on elections. *Journal of Artificial Intelligence Research, 40,* 305–351.
24. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2009). Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research, 35,* 275–341.
25. Faliszewski, P., & Rothe, J. (2016). Control and bribery in voting. In: F. Brandt, V. Conitzer, U. Endriss, J. Lang, A. Procaccia (eds.) Handbook of Computational Social Choice, chap. 7, pp. 146–168. Cambridge University Press

26. Gabow, H. (1983). An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In: Proceedings of the 15th ACM Symposium on Theory of Computing, pp. 448–456
27. Garey, M., & Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman and Company: W. H.
28. Ghosh, S., Mundhe, M., Hernandez, K., & Sen, S. (1999). Voting for movies: The anatomy of recommender systems. In: Proceedings of the 3rd Annual Conference on Autonomous Agents, pp. 434–435
29. Gonzalez, T. (1985). Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science, 38,* 293–306.
30. Grötschel, M., Lovász, L., & Schrijver, A. (1988). *Geometric Algorithms and Combinatorial Optimization*. Berlin: Springer.
31. Guo, J., & Shrestha, Y.R. (2014). Controlling two-stage voting rules. In: Proceedings of the 21st European Conference on Artificial Intelligence, pp. 411–416
32. 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, pp. 308–315
33. Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2007). Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence, 171*(5–6), 255–285.
34. Hemaspaandra, E., Hemaspaandra, L., & Schnoor, H. (2014). A control dichotomy for pure scoring rules. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence, pp. 712–720
35. Hemaspaandra, E., Hemaspaandra, L. A., & Menton, C. (2020). Search versus decision for election manipulation problems. *ACM Transactions on Computation Theory, 12*(1), 3:1-3:42.
36. Hemaspaandra, L. (2018). Computational social choice and computational complexity: BFFs? In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp. 7971–7977
37. Karp, R. (1972). Reducibility among combinatorial problems. In: R. Miller, J. Thatcher (eds.) Complexity of Computer Computations, pp. 85–103
38. Lang, J., Maudet, N., & Polukarov, M. (2013). New results on equilibria in strategic candidacy. In: Proceedings of the 6th International Symposium on Algorithmic Game Theory, pp. 13–25
39. Lin, A. (2011). The complexity of manipulating *k*-approval elections. In: Proceedings of the 3rd International Conference on Agents and Artificial Intelligence, pp. 212–218
40. Loreggia, A. (2012). Iterative voting and multi-mode control in preference aggregation. Master's thesis, University of Padova
41. Loreggia, A. (2014). Iterative voting and multi-mode control in preference aggregation. *Intelligenza Artificiale, 8*(1), 39–51.
42. Loreggia, A. (2016). Iterative voting, control and sentiment analysis. Ph.D. thesis, University of Padova
43. Loreggia, A., Narodytska, N., Rossi, F., Venable, B., & 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, pp. 1737–1738
44. Magiera, K., & Faliszewski, P. (2017). How hard is control in single-crossing elections? *Journal of Autonomous Agents and Multi-Agent Systems, 31*(3), 606–627.
45. Maushagen, C., & 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, pp. 277–285
46. Maushagen, C., & 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.
47. Maushagen, C., & Rothe, J. (2020). The last voting rule is home: Complexity of control by partition of candidates or voters in maximin elections. In: Proceedings of the 24th European Conference on Artificial Intelligence, pp. 163–170
48. Menton, C. (2013). Normalized range voting broadly resists control. *Theory of Computing Systems, 53*(4), 507–531.
49. Menton, C., & Singh, P. (2013). Control complexity of Schulze voting. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, pp. 286–292
50. Neveling, M., Rothe, J., & Zorn, R. (2020). The complexity of controlling Condorcet, fallback, and *k*-veto elections by replacing candidates or voters. In: Proceedings of the 15th International Computer Science Symposium in Russia, pp. 314–327
51. Niedermeier, R. (2006). *Invitation to Fixed-Parameter Algorithms*. Oxford: Oxford University Press.

52. Obraztsova, S., Elkind, E., & Hazon, N. (2011). Ties matter: Complexity of voting manipulation revisited. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 2698–2703
53. Oflazer, K., & Tür, G. (1997). Morphological disambiguation by voting constraints. In: Proceedings of the *8th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 222–229
54. Parkes, D., & Xia, L. (2012). A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence, pp. 1429–1435
55. Pennock, D., Horvitz, E., & Giles, C. (2000). Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In: Proceedings of the 17th National Conference on Artificial Intelligence, pp. 729–734
56. Rothe, J. (2005). Complexity Theory and Cryptology. An Introduction to Cryptocomplexity. EATCS Texts in Theoretical Computer Science. Springer-Verlag
57. Sigletos, G., Paliouras, G., Spyropoulos, C., & Hatzopoulos, M. (2005). Combining information extraction systems using voting and stacked generalization. *Journal of Machine Learning Research, 6,* 1751–1782.
58. Tovey, C. (2002). Tutorial on computational complexity. *Interfaces, 32*(3), 30–61.
59. West, D. (2000). *Introduction to Graph Theory*. New Jersey: Prentice-Hall.
60. Yang, Y. (2017). The complexity of control and bribery in majority judgment. In: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems, pp. 1169–1177
61. Yang, Y., & Guo, J. (2014).Controlling elections with bounded single-peaked width. In: Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems, pp. 629–636
62. Yang, Y., & Guo, J. (2017). The control complexity of *r*-approval: From the single-peaked case to the general case. *Journal of Computer and System Sciences, 89,* 432–449.
63. Yang, Y., & Wang, J. (2017). Anyone but them: The complexity challenge for a resolute election controller. In: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems, pp. 1133–1141

Authors and Affiliations

Gábor Erdélyi[1] · Marc Neveling[2] · Christian Reger[3] · Jörg Rothe[2] · Yongjie Yang[4] · Roman Zorn[2]

Gábor Erdélyi
gabor.erdelyi@canterbury.ac.nz

Marc Neveling
marc.neveling@hhu.de

Christian Reger
christian.reger@ymail.com

Jörg Rothe
rothe@hhu.de

Roman Zorn
roman.zorn@hhu.de

[1] School of Mathematics and Statistics, University of Canterbury, Christchurch, New Zealand

[2] Institut für Informatik, Heinrich-Heine-Universität Düsseldorf, Düsseldorf, Germany

[3] School of Economic Disciplines, University of Siegen, Siegen, Germany

[4] Chair of Economic Theory, Saarland University, Saarbrücken, Germany