Journal of Artificial Intelligence Research 55 (2016) 603-652 Submitted 07/15; published 03/16

Large-Scale Election Campaigns: Combinatorial Shift Bribery

Robert Bredereck ROBERT.BREDERECK@TU-BERLIN.DE
TU Berlin,
Berlin, Germany

Piotr Faliszewski FALISZEW@AGH.EDU.PL
AGH University of Science and Technology,
Kraków, Poland

Rolf Niedermeier ROLF.NIEDERMEIER@TU-BERLIN.DE
Nimrod Talmon NIMRODTALMON77@GMAIL.COM
TU Berlin,
Berlin, Germany

Abstract

We study the complexity of a combinatorial variant of the Shift Bribery problem in elections. In the standard Shift Bribery problem, we are given an election where each voter has a preference order over the set of candidates and where an outside agent, the briber, can pay each voter to rank the briber's favorite candidate a given number of positions higher. The goal is to ensure the victory of the briber's preferred candidate. The combinatorial variant of the problem, introduced in this paper, models settings where it is possible to affect the position of the preferred candidate in multiple votes, either positively or negatively, with a single bribery action. This variant of the problem is particularly interesting in the context of large-scale campaign management problems (which, from the technical side, are modeled as bribery problems). We show that, in general, the combinatorial variant of the problem is highly intractable; specifically, NP-hard, hard in the parameterized sense, and hard to approximate. Nevertheless, we provide parameterized algorithms and approximation algorithms for natural restricted cases.

1. Introduction

We study the computational complexity of election campaign management for the case where campaign actions (such as airing a TV advertisement, launching a web-based campaign, or organizing meetings with voters) may have large-scale effects which affect multiple voters. Further, we are interested in settings where these actions can have both positive effects (for example, some voters may choose to rank the promoted candidate higher because they find arguments presented in a given advertisement appealing) as well as negative ones (for example, because some other voters find the advertisement to be too aggressive). Thus, in our setting, the two major issues faced by a campaign manager are (a) choosing actions

that positively affect as many voters as possible and (b) balancing the negative effects of campaigning actions (for example, by concentrating these negative effects on voters who disregard the promoted candidate anyway).

Our research falls within the field of computational social choice, a subarea of multiagent systems. We use the standard election model, where we are given a set C of candidates and a collection V of voters, each represented by her preference order (that is, a ranking of the candidates from the most preferred one to the least preferred one). We assume that we know the preferences of all the voters. While having such perfect knowledge is impossible in practice, this assumption is a convenient simplification that models the fact that we may have (approximate) information from preelection polls or some other sources.

We consider two voting rules, the Plurality rule (where we pick the candidate who is ranked first by most voters) and the Borda rule (where each candidate c gets from each voter v as many points as there are candidates that v prefers c to, and we pick the candidate with the most points). These rules are chosen because the Plurality rule is the most widespread rule in practice and because the Borda rule is very well-studied in the context of campaign management.

Within computational social choice, the term *campaign management* (introduced in Elkind, Faliszewski, & Slinko, 2009; Elkind & Faliszewski, 2010) is an alternative name for the bribery family of problems (introduced in Faliszewski, Hemaspaandra, & Hemaspaandra, 2009a) for the cases where one focuses on modeling actions available during election campaigns: As a result of money spent by a campaign manager, some of the voters change their votes. In this paper we study campaign management through the Shift Bribery problem (Elkind et al., 2009; Elkind & Faliszewski, 2010; Bredereck, Chen, Faliszewski, Nichterlein, & Niedermeier, 2014a; Bredereck, Faliszewski, Niedermeier, & Talmon, 2016). In Shift Bribery we have a candidate p who we want to win, for each voter v we have a price $\pi_v(i)$ for which this voter is willing to shift p forward by i positions in her preference order[1], and we ask for the lowest cost of ensuring that p is a winner (see Section 1.1 for references to other campaign management problems).

The Shift Bribery problem has one major drawback as a model for campaign management. It is incapable of capturing large-scale effects of campaign actions. In particular, if one puts forward a TV spot promoting a given candidate, then some voters will react positively and rank the candidate higher, some will be oblivious to it, and some will react negatively, by ranking the candidate lower. Shift Bribery cannot model such correlated effects. In this paper we introduce and study the Combinatorial Shift Bribery problem, allowing campaign actions to have effects, positive or negative, on whole groups of voters.

We are interested in understanding how a more realistic model of campaign management affects the complexity of the problem. Indeed, Shift Bribery is, computationally, a very well-behaved problem. For example, for the Plurality rule it is solvable in polynomial time and for the Borda rule it is NP-complete (Elkind et al., 2009), but there is a polynomial-time 2-approximation algorithm (Elkind et al., 2009; Elkind & Faliszewski, 2010) and there are fixed-parameter (FPT) algorithms, either exact or capable of finding solutions arbitrarily close to the optimal ones (Bredereck et al., 2014a). In this work, we ask to what extent

1. Of course, this price does not necessarily reflect a direct money transfer to the voter, but rather the cost of convincing the voter to change his or her mind.

do we retain these good computational properties when we allow large-scale effects. The results are surprising both positively and negatively:

1. Combinatorial Shift Bribery becomes both NP-complete and W[1]-hard even for the Plurality rule, even for very restrictive choice of parameters, even if the correlated effects of particular campaign actions are limited to at most two voters. Moreover, our hardness results imply that good, general approximation algorithms do not exist when we allow negative effects of campaign actions.

2. In spite of the above, it is still possible to derive relatively good (approximation) algorithms, both for the Plurality rule and for the Borda rule, provided that we restrict the effects of the campaign actions to be only positive and to either only involve few voters each, or to only involve groups of consecutive voters (with respect to an ordering over the voters which might correspond, for example, to time).

Our results are summarized in Table 1 in Section 4. With the generality of our problem and its combinatorial nature it is natural that we obtain many hardness results. Yet, their extent and strength is surprising, and so is the fact that we also find a nontrivial landscape of tractable cases.

1.1 Related Work

Our work builds on top of two main research ideas. First, on studying campaign management/bribery problems, and, second, on studying combinatorial variants of election problems.

The study of the computational complexity of bribery in elections was initiated by Faliszewski et al. (2009a), and continued by a number of researchers (Faliszewski, Hemaspaandra, Hemaspaandra, & Rothe, 2009b; Hazon, Lin, & Kraus, 2013; Mattei, Goldsmith, & Klapper, 2012a; Mattei, Pini, Rossi, & Venable, 2012b). Elkind et al. (2009) and Elkind and Faliszewski (2010) realized that the formalism of election bribery problems is useful from the point of view of planning election campaigns. In particular, they defined the Swap Bribery problem and its restricted variant, Shift Bribery. In the former it is possible, at a given price, to swap any two adjacent candidates in a given vote. In the latter, we are only allowed to shift the preferred candidate forward. Various problems, modeling different flavors of campaign management, have been studied, including, for example, the possibility to alter the number of approved/ranked candidates (Baumeister, Faliszewski, Lang, & Rothe, 2012; Faliszewski, Reisch, Rothe, & Schend, 2014; Schlotter, Faliszewski, & Elkind, 2011). Different (positive) applications of bribery problems include, for example, the Margin of Victory problem, where the goal of the briber is to prevent some candidate from winning. If it is possible to do so at low cost, then this suggests that the election could have been tampered with (Cary, 2011; Magrino, Rivest, Shen, & Wagner, 2011; Xia, 2012; Reisch, Rothe, & Schend, 2014).

From our point of view, the most related works are those of Elkind et al. (2009), Elkind and Faliszewski (2010), Bredereck et al. (2014a, 2016), and Dorn and Schlotter (2012). The former ones study Shift Bribery, which we generalize (parameterized complexity of Shift Bribery is studied in Bredereck et al., 2014a, while Shift Bribery for multiwinner

elections are studied in Bredereck et al., 2016), whereas the work of Dorn and Schlotter (2012) pioneers the use of parameterized complexity analysis for (swap) bribery problems.

Our work is largely inspired by that of Bulteau, Chen, Faliszewski, Niedermeier, and Talmon (2015) and Chen, Faliszewski, Niedermeier, and Talmon (2015), who introduced and studied combinatorial variants of election control. Election control is a very well-studied topic in computational social choice, initiated by Bartholdi, Tovey, and Trick (1992) and then studied by numerous researchers (we point the readers to Faliszewski, Hemaspaandra, & Hemaspaandra, 2010; Faliszewski & Rothe, 2015, for a detailed account). Briefly put, control problems model attempts at changing the election results by changing their structure. The standard types of control include adding, deleting, and partitioning candidates or voters. Control problems, especially those related to adding and deleting voters, are quite relevant to the issues of campaign management, and, indeed, in Section 5 we do show a connection between Combinatorial Shift Bribery and (combinatorial) control by adding voters (Bulteau et al., 2015).

The idea of combinatorial shift bribery is somewhat related to the problem of lobbying in multiple referenda, as introduced by Christian, Fellows, Rosamond, and Slinko (2007) (parameterized study was provided in Bredereck, Chen, Hartung, Kratsch, Niedermeier, Suchý, & Woeginger, 2014b; probabilistic variant was studied, also in the parameterized sense, in Binkele-Raible, Erdélyi, Fernau, Goldsmith, Mattei, & Rothe, 2014). There, we have a number of yes/no elections and the goal is to ensure that for each of these election a majority of the voters vote "yes." Each single lobbying action can convince one voter to vote "yes" in all the elections. In combinatorial shift bribery we have a single election and a single action can affect multiple voters, whereas in the lobbying problem we have multiple elections but each action affects only one voter.

We stress that our use of the term "combinatorial variants of election problems" is different than the one used in the well-established line of work regarding combinatorial candidate spaces (see Lang & Xia, 2015, and further works, for example, Boutilier, Brafman, Hoos, & Poole, 2004; Conitzer, Lang, & Xia, 2009; Mattei et al., 2012b). In our work we use the term "combinatorial" to refer to the "combinations" of voters affected by each bribery action.

1.2 Organization of the Paper

After providing some preliminaries in Section 2, we give the formal definition of the Combinatorial Shift Bribery problem in Section 3. In Section 4 we give an overview of our results. We shed light on some connections between Combinatorial Shift Bribery and the problem of Combinatorial Control in Section 5. Then, in Section 6, we present a series of strong hardness results covering all our classes of shift actions for very restrictive sets of parameters (for example, many of our results already apply to the case of two candidates). In Section 7, we develop several exact algorithms for special cases of Combinatorial Shift Bribery, while in Section 8 we describe our approximation algorithms for Combinatorial Shift Bribery. Some of our proofs are available in the appendices (either when a given proof relies on ideas already presented in other proofs, or—as in the case of Theorem 9—when the proof is particularly involved). We end with conclusions in Section 9.

2. Preliminaries

In this section, we briefly describe our model of elections, define the two voting rules that we study, and review basic concepts from parameterized complexity.

2.1 Elections

An election $E = (C, V)$ consists of a set $C = \{c_1, \ldots, c_m\}$ of candidates and of a collection $V = (v_1, \ldots, v_n)$ of voters. Each voter is represented through her preference order, that is, a linear ranking of the candidates from the most preferred one to the least preferred one; we use voters and preference orders interchangeably. For example, if $C = \{c_1, c_2, c_3\}$, then voter v_1 may have preference order $v_1 \colon c_1 \succ c_2 \succ c_3$ to indicate that she likes c_1 best, then c_2, and then c_3 (for clarity, we treat the voters as females and the candidates as males).

We assume that there is an arbitrary (but fixed) canonical order over the set of candidates (for example, one could order the candidates lexicographically by their names). For a subset $A \subseteq C$ of candidates, writing $\overrightarrow{A}$ within a preference order means listing the candidates from A in this canonical order, and writing $\overleftarrow{A}$ means listing them in the reverse of this order.

2.2 Voting Rules

A voting rule $\mathcal{R}$ is a function that, given an election $E = (C, V)$, outputs a set $\mathcal{R}(E) \subseteq C$ of (tied) election winners. Each candidate $c \in \mathcal{R}(E)$ is said to be an *$\mathcal{R}$-winner* of the election E. We consider two election rules, the Plurality rule and the Borda rule. Both assign points to candidates and output those with the highest score. Under the Plurality rule, each candidate receives one point for each voter that ranks him first. Under the Borda rule, each candidate receives i points for each voter that prefers this candidate to exactly i other ones.

We use the nonunique-winner model. That is, all the candidates selected by a given voting rule are viewed as equally successful winners (in practice, of course, one has to use some sort of a tie-breaking rule to resolve the situation, but disregarding ties simplifies the analysis; however, an interested reader should consult papers on the effects of tie-breaking on the complexity of election problems, e.g. Obraztsova & Elkind, 2011; Obraztsova, Elkind, & Hazon, 2011).

2.3 Parameterized Complexity

We assume familiarity with standard notions regarding algorithms and complexity theory, but briefly review notions regarding parameterized complexity theory (Downey & Fellows, 2013; Flum & Grohe, 2006; Niedermeier, 2006).

In parameterized complexity theory we measure the complexity of a given problem with respect to both the input size and a particular *parameter* of the problem. Typical parameters for election problems include the number of candidates, the number of voters, and the solution size (for example, the number of campaign actions one can perform; see Betzler, Bredereck, Chen, & Niedermeier, 2012, for a survey of parameterized complexity and voting). We say that a parameterized problem is *fixed-parameter tractable* (is in FPT) if there is an algorithm that given an input instance I with parameter k solves the problem in

$g(k)|I|^{O(1)}$ time, where g is some computable function and $|I|$ is the length of the encoding of I. There is also a hierarchy of hardness classes for parameterized problems, of which the two most important levels are formed by the classes W[1] and W[2]. The most convenient way of defining these classes is through an appropriate reduction notion and their complete problems. Specifically, we say that a parameterized problem A reduces to a parameterized problem B if there are two computable functions, h and h', with the following properties: given an instance I of A with parameter k, $h(I)$ outputs in FPT time (i.e., in time $g(k)|I|^{O(1)}$ for some computable function g) an instance I' of B with parameter $k' \leq h'(k)$, such that I is a "yes"-instance of A if and only if I' is a "yes"-instance of B. In other words, h is a many-one reduction from A to B that is allowed to run in FPT time, but that is required to output an instance whose parameter is upper-bounded by a function of the input instance's parameter.

The class W[1] is defined as the class of problems that parameterically reduce to the Clique problem, and W[2] as the class of problems that parameterically reduce to the Set Cover problem, where both problems are parameterized by the solution size (that is, by the value h from their definitions).

Clique
Input: An undirected graph $G = (V(G), E(G))$ and an integer h.
Question: Is there a set H of h vertices such that there is an edge between each pair of vertices from H?

Set Cover
Input: A universe set X, a family $\mathcal{S}$ of subsets of X, and an integer h.
Question: Is there a subset $\mathcal{S}' \subseteq \mathcal{S}$ of at most h subsets whose union gives X?

We sometimes consider special variants of these problems that we describe in detail within relevant proofs.

A parameterized problem is contained in the class XP if there is an algorithm that, given an instance I for it with parameter k, solves it in time $|I|^{g(k)}$, where g is some computable function. It holds that FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq$ XP. We point the readers interested in further details regarding parameterized complexity theory (and the design of parameterized algorithms) to the textbooks of Downey and Fellows (2013), Flum and Grohe (2006), and Niedermeier (2006).

3. The Combinatorial Shift Bribery Problem

In this section we first define the Combinatorial Shift Bribery problem in its full generality and, then, we describe why and how we simplify it for the remainder of our study.

3.1 The Definition

Let $\mathcal{R}$ be some voting rule. The definition of $\mathcal{R}$-Combinatorial Shift Bribery is somewhat involved, therefore we first define some necessary components. We are given an election $E = (C, V)$ and a *preferred candidate* $p \in C$. The goal is to ensure that p is an $\mathcal{R}$-winner of the election. To this end, we have a number of possible actions to choose from.

Let $m := |C|$ be the number of candidates in E and let $n := |V|$ be the number of voters. A shift action f is an n-dimensional vector of (possibly negative) integers, $f = (f^{(1)}, \ldots, f^{(n)})$. In $\mathcal{R}$-Combinatorial Shift Bribery we are given a family $F = (f_1, \ldots, f_\zeta)$ of shift actions. Each particular shift action models a possible campaigning action, such as airing a TV spot or organizing a meeting with the voters. The components of a given shift action measure the effects of this action on the particular voters. For a given subset $F' \subseteq F$ of available shift actions, we define the effect of F' on voter v_i $(1 \leq i \leq n)$ as $\mathcal{E}^{(i)}(F') = \sum_{f_j \in F'} f_j^{(i)}$. Further, each shift action f_j $(1 \leq j \leq \zeta)$ comes with a nonnegative integer cost $w(f_j)$ for its application.

Each voter v_i $(1 \leq i \leq n)$ has her individual *threshold function* $\pi_i : \mathbb{Z} \to \mathbb{Z}$ describing how shift actions affect this voter. We require that $\pi_i(0) = 0$ and that π_i is nondecreasing. Let F' be a collection of shift actions. After applying the shift actions from F', each voter v_i $(1 \leq i \leq n)$ shifts the preferred candidate p by $t > 0$ positions forward if (a) $\mathcal{E}^{(i)}(F') > 0$, and (b) $\pi_i(t) \leq \mathcal{E}^{(i)}(F') < \pi_i(t+1)$. The shift is by $t > 0$ positions back if (a) $\mathcal{E}^{(i)}(F') < 0$, and (b) $\pi_i(-t) \geq \mathcal{E}^{(i)}(F') > \pi_i(-t-1)$.

Finally, we are given a nonnegative integer B, the *budget*. We ask for the existence of a collection $F' \subseteq F$ of available shift actions with total cost $\sum_{f_j \in F'} w(f_j)$ at most B and such that after applying them p is an $\mathcal{R}$-winner of the given election. If this is the case, then we say that F' is *successful*. Consider the following example.

Example 1. Consider the election below, where the set of candidates is $C = \{a, b, c, p\}$, the collection of voters is $V = (v_1, v_2, v_3)$, and p is the preferred candidate. There are three available shift actions, each with the same unit cost (i.e., $w(f_1) = w(f_2) = w(f_3) = 1$).

election	**shift actions**		
$v_1 : c \succ b \succ p \succ a$	$\begin{pmatrix} 2 \\ 4 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 6 \\ 0 \\ -3 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix}$
$v_2 : b \succ a \succ c \succ p$			
$v_3 : p \succ a \succ b \succ c$			
	f_1	f_2	f_3

The threshold functions are such that:

1. $\pi_1(-1) = -4$, $\pi_1(0) = 0$, $\pi_1(1) = 6$, $\pi_1(2) = 100$.
2. $\pi_2(0) = 0$, $\pi_2(1) = 2$, $\pi_2(2) = \pi_2(3) = 100$.
3. $\pi_3(-3) = \pi_3(-2) = -100$, $\pi_3(-1) = -3$, $\pi_3(0) = 0$.

We use the Borda rule. Candidates a, b, c, and p have, respectively, 4, 6, 4, and 4 points. It is easy to see that applying any single shift action does not ensure p's victory. However, applying shift actions $F' = \{f_2, f_3\}$ results in p being a winner. The total effect of these two shift actions is $(6, 2, -3)$. According to the threshold functions, this means that p is shifted forward by one position in v_1 and v_2, and is shifted back by one position in v_3. After these shifts, the modified election looks as follows:

election

$$v_1' : c \succ p \succ b \succ a$$
$$v_2' : b \succ a \succ p \succ c$$
$$v_3' : a \succ p \succ b \succ c$$

That is, after we apply the shift actions $F' = \{f_2, f_3\}$, we have that candidate c has 3 points, while all other candidates have 5 points each. Thus, a, b, and p are tied as winners and F' is indeed a successful set of shift actions. △

Formally, given a voting rule $\mathcal{R}$, we define the $\mathcal{R}$-Combinatorial Shift Bribery problem as follows:

$\mathcal{R}$-Combinatorial Shift Bribery
Input: An election $E = (C, V)$, where $C = \{c_1, \ldots, c_m\}$ is the set of candidates and $V = (v_1, \ldots, v_n)$ is the collection of voters, a set $F = \{f_1, \ldots, f_\zeta\}$ of shift actions with costs $w(f_1), \ldots, w(f_\zeta)$, threshold functions $\pi_1, \ldots, \pi_n$, and a nonnegative integer budget B. One of the candidates is designated as the preferred candidate p.
Question: Is there a subset $F' \subseteq F$ of shift actions with total cost at most B such that after we apply the shift actions from F' candidate p is an $\mathcal{R}$-winner of the resulting election?

While this definition is quite complicated, it captures some important features of campaigning. For example, the use of threshold functions allows us to model voters who are unwilling to change the position of the preferred candidate beyond a certain range, irrespective of the strength of the campaign. The fact that different shift actions have different costs models the fact that particular actions (for example, airing TV spots or organizing meetings) may come at different costs.

3.2 Relation to Standard Shift Bribery

It is necessary to comment on the relation between our Combinatorial Shift Bribery problem and its non-combinatorial variant, Shift Bribery (Elkind et al., 2009; Elkind & Faliszewski, 2010).

The non-combinatorial variant of the Shift Bribery problem is defined very similarly to the combinatorial one, but the voters have no threshold functions and instead of the collection of shift actions and their costs, each voter v_i has his or her shift-bribery price function ρ_i. The cost of shifting the preferred candidate forward by t positions in v_i's preference order is $\rho_i(t)$ (only forward shifts are allowed). We require that $\rho_i(0) = 0$ and that the functions are nondecreasing. Formally, we have the following definition ($\mathcal{R}$ is a voting rule).

$\mathcal{R}$-Shift Bribery
Input: An election $E = (C, V)$, where $C = \{c_1, \ldots, c_m\}$ is the set of candidates and $V = (v_1, \ldots, v_n)$ is the collection of voters, a collection $(\rho_1, \ldots, \rho_n)$ of shift-bribery price functions, and a nonnegative integer budget B. One of the

candidates is designated as the preferred candidate p.
Question: Is there a vector $(s_1, \ldots, s_n)$ of natural numbers such that (a) $\sum_{i=1}^{n} \rho_i(s_i) \leq B$ and (b) if for each voter v_i we shift p forward by s_i positions, then p is an $\mathcal{R}$-winner of the resulting election?

While intuitively it seems that $\mathcal{R}$-Shift Bribery is simpler than its combinatorial cousin, making this observation formal requires some care.

Proposition 1. *Let $\mathcal{R}$ be a voting rule. It holds that $\mathcal{R}$-*Shift Bribery *many-one reduces to $\mathcal{R}$-*Combinatorial Shift Bribery *in polynomial time.*

Proof. Consider an instance of $\mathcal{R}$-Shift Bribery with an election $E = (C, V)$, where $C = \{c_1, \ldots, c_m\}$ and $V = (v_1, \ldots, v_n)$, with a collection of shift-bribery price functions $(\rho_{v_1}, \ldots, \rho_{v_n})$, and with budget B. Without loss of generality, we take c_1 to be the preferred candidate and denote him or her as p. We form an instance of $\mathcal{R}$-Combinatorial Shift Bribery with the same election, the same budget, and the same preferred candidate, but where the shift actions, their costs, and voters' threshold functions are constructed as follows: for each voter v_i, we set his or her threshold function to be $\pi_i(t) = \sum_{j=1}^{t} 2^{m-j}$, and for each number t of positions by which it is possible to shift the preferred candidate forward in v_i's preference order, we create a shift action $f_{i,t}$ that has a zero effect on all voters but v_i, on whom it has effect 2^{m-t}; the cost of $f_{i,t}$ is $w(f_{i,t}) = \rho(t) - \rho(t-1)$.

If there is a sequence $(s_1, \ldots, s_n)$ such that $\sum_{i=1}^{n} \rho_i(s_i) \leq B$ and p is an $\mathcal{R}$-winner of the election where for each voter v_i we shift p forward by s_i positions, then there is also a solution for our constructed instance of Combinatorial Shift Bribery: if for each v_i, we use shift actions $f_{i,1}, \ldots, f_{i,s_i}$, then the total bribery cost is the same as in the Shift Bribery instance and, after implementing the shifts, for each v_i the preferred candidate is shifted by exactly s_i positions.

Now assume that our constructed Combinatorial Shift Bribery instance is a "yes"-instance. Consider some subset F' of shift actions whose total cost is at most B and which ensure that p is the $\mathcal{R}$-winner of the election (and recall that each shift action can be used at most once). For each voter $v_i \in V$, we define s_i to be the largest integer such that shift actions $f_{i,1}, \ldots, f_{i,s_i}$ all belong to F'. Let us fix some voter v_i. We claim that after applying the shift actions from F' (in the Combinatorial Shift Bribery instance), the preferred candidate is shifted forward by exactly s_i positions. By the definition of s_i, it is immediate that he or she is shifted forward by at least s_i positions. He or she is not shifted forward by more positions for the following reason: the shift actions $f_{i,1}, \ldots, f_{i,s_i}$ have total effect on v_i equal to $\sum_{j=1}^{s_i} 2^{m-j}$, which is equal to $\pi_i(s_i)$. By definition, shift action f_{i,s_i+1} is not in F'. The sum of all the remaining shift actions that have an effect on v_i is smaller than:

$$\sum_{j=s_i+2}^{m} 2^{m-j} = \sum_{j=0}^{m-s_i-2} 2^j = 2^{m-s_i-1} - 1.$$

However, $\pi_i(s_i+1) - \pi_i(s_i) = 2^{m-s_i-1}$. This means that even if we used all the shift actions aside from f_{i,s_i+1}, in v_i's preference order we still would shift p by exactly s_i positions.

In conclusion, this means that implementing the shift actions F' ensures that for each voter v_i we shift p forward by exactly s_i positions. Further, for each v_i we have that

$w(f_{i,1}) + \cdots + w(f_{i,s_i}) = \rho_i(s_i)$. Therefore, the sequence $(s_1, \ldots, s_n)$ witnesses that the input instance of Shift Bribery is a "yes"-instance because the total cost of the shifts is at most B (as in the combinatorial instance) and they ensure that p is a winner (as in the combinatorial instance).

Since the reduction clearly runs in polynomial time, the proof is complete. □

The construction from the above proof is somewhat involved, especially if one takes into account that it simply shows that our Combinatorial Shift Bribery problem indeed generalizes the much simpler, non-combinatorial, one. Nonetheless, its somewhat contrived use of threshold functions seems to be necessary. Indeed, if in the Combinatorial Shift Bribery problem we restricted the shift actions to have positive entries for exactly one voter each, and we used simple linear threshold functions, then we would obtain Shift Bribery for the case of *convex price functions* (Bredereck et al., 2014a). This is a very general variant of the Shift Bribery problem for which, for example, all the NP-hardness results of Elkind et al. (2009) hold (as shown in Bredereck et al., 2014a), but nonetheless not the most general one.

3.3 A General Hardness Result

It turns out that the Combinatorial Shift Bribery problem, as defined in Section 3.1 above, is so general that it allows for the following, sweeping, hardness result.[2]

Theorem 2. *For both the Plurality rule and the Borda rule,* Combinatorial Shift Bribery *is* NP*-hard even for five voters and two candidates and no budget constraints. For the Borda rule,* Combinatorial Shift Bribery *is* NP*-hard also for three voters and four candidates.*

Proof. We reduce from the following (weakly NP-hard) variant of the Subset Sum problem (it is a simple exercise to show its NP-hardness through a reduction from the classic Subset Sum problem):

> Subset Sum (Zero Variant)
> **Input:** A set $A := \{a_1, \ldots, a_n\}$ of integers.
> **Question:** Is there a nonempty set $A' \subseteq A$ such that $\sum_{a_i \in A'} a_i = 0$?

Given an instance $A = \{a_1, \ldots, a_n\}$ of Subset Sum (Zero Variant), we construct an instance of Plurality-Combinatorial Shift Bribery with two candidates. Since the Plurality rule and the Borda rule coincide for elections with two candidates, our hardness result transfers to Borda-Combinatorial Shift Bribery (and, in fact, to almost all natural voting rules).

We construct the following election:

2. Note, however, that we prove weak NP-hardness. That is, our result may not hold if we assume that all occurring numbers are encoded in unary. On the contrary, all other hardness proofs in this paper give strong hardness results and are independent of such number encoding issues.

$$\begin{array}{lcccc} \textbf{election} & & \textbf{shift actions} & & \\ v_1 : p \succ d & \begin{pmatrix} a_1 \end{pmatrix} & & & \begin{pmatrix} a_n \end{pmatrix} \\ v_2 : p \succ d & \begin{pmatrix} -a_1 \end{pmatrix} & & & \begin{pmatrix} -a_n \end{pmatrix} \\ v_3 : d \succ p & \begin{pmatrix} 1 \end{pmatrix} & \cdots & & \begin{pmatrix} 1 \end{pmatrix} \\ v_4 : d \succ p & \begin{pmatrix} 0 \end{pmatrix} & & & \begin{pmatrix} 0 \end{pmatrix} \\ v_5 : d \succ p & \begin{pmatrix} 0 \end{pmatrix} & & & \begin{pmatrix} 0 \end{pmatrix} \\ & f_1 & \cdots & & f_n \end{array}$$

That is, for each element $a_i \in A$, the set F of shift actions contains one shift action f_i with effect a_i on v_1, effect $-a_i$ on v_2, effect 1 on v_3, and no effect on the other two voters. The voter threshold functions are as follows. Candidate p is shifted to the last position for v_1 and v_2 if the effect on these voters is negative (that is, $\pi_1(-1) = \pi_2(-1) = -1$). Candidate p is shifted to the top position for the third voter if the effect is positive (that is, $\pi_3(1) = 1$). We set the cost of each shift action to be one and we set our budget to be n. Thus the budget allows us to pick any combination of the shift actions.

For the "if" direction, let $A' \subseteq A$ be a non-empty subset whose element-wise sum equals zero. After applying $F' := \{f_i \mid a_i \in A'\}$, p is a winner: Since A' sums up to zero, there is no effect on the first two voters. The effect on the third voter is positive, because A' is non-empty. Thus p is preferred by three of the five voters and wins the election.

For the "only if" direction, let $F' \subseteq F$ be a subset of shift actions that makes p a winner. Then, F' must be non-empty because p does not win the initial election. We claim that the element-wise sum of $A' := \{a_i \mid f_i \in F'\}$ is zero. For the sake of contradiction, assume that $\sum_{a_i \in A'} a_i \neq 0$. If the sum were negative, then there would be a negative effect on the first voter, d would be preferred by three voters out of five, and d would win the election. If the sum were positive, then we would have the same effect with the second voter taking the role of the first one.

Using a very similar idea, we can show how to reduce Subset Sum (Zero Variant) to Borda-Combinatorial Shift Bribery with three voters and four candidates. Given the input as before, we construct the following instance:

$$\begin{array}{lcccc} \textbf{election} & & \textbf{shift actions} & & \\ v_1 : p \succ d_1 \succ d_2 \succ d_3 & \begin{pmatrix} 3a_1 \end{pmatrix} & & & \begin{pmatrix} 3a_n \end{pmatrix} \\ v_2 : p \succ d_1 \succ d_2 \succ d_3 & \begin{pmatrix} -3a_1 \end{pmatrix} & \cdots & & \begin{pmatrix} -3a_n \end{pmatrix} \\ v_3 : d_1 \succ d_2 \succ d_3 \succ p & \begin{pmatrix} 3 \end{pmatrix} & & & \begin{pmatrix} 3 \end{pmatrix} \\ & f_1 & \cdots & & f_n \end{array}$$

That is, for each element $a_i \in A$, F contains one shift action f_i with effect $3a_i$ on v_1, effect $-3a_i$ on v_2, and effect 3 on v_3. Each voter v_i has the same threshold function $\pi_i(t) = t$. In effect, p is shifted to the last position of the first and of the second voter if the effect on these voters is negative, and is shifted to the top position of the third vote if the effect there is positive. Each shift action has the same unit cost, and we set the budget to n (i.e., we can pick any combination of the shift actions).

Observe that d_1 is the original winner of the election and obtains seven points whereas p obtains only six points.

For the "if" direction, let $A' \subseteq A$ be a non-empty subset whose element-wise sum equals zero. If we apply shift actions $F' := \{f_i \mid a_i \in A'\}$ then p becomes a winner: Since A' sums up to zero, there is no effect on the first two voters. The effect on the third voter is positive because A' is non-empty. Thus, p is the most preferred candidate for all the voters and wins the election.

For the "only if" direction, let $F' \subseteq F$ be a subset of shift actions that makes p a winner. Then, F' must be non-empty because p does not win the initial election. We show that the element-wise sum of $A' := \{a_i \mid f_i \in F'\}$ is zero. For the sake of contradiction assume that $\sum_{a_i \in A'} a_i \neq 0$. If the sum were negative, then there would be a negative effect on the first voter and p would obtain six points, whereas d_1 would obtain seven. If the sum were positive, we would have the same effect with the roles of the first and the second voter switched. □

Effectively, Theorem 2 shows that studying large-scale effects of campaign actions through the full-fledged $\mathcal{R}$-Combinatorial Shift Bribery problem leads to a hopelessly intractable problem: We have hardness even for elections with both a fixed number of candidates and a fixed number of voters.

3.4 Restricted Variants of Combinatorial Shift Bribery

Given the hardness results from Theorem 2, throughout the remainder of the paper we focus on restricted variants of the Combinatorial Shift Bribery problem. We assume the individual threshold functions to be the identity functions (that is, for each voter i and each integer t, it holds that $\pi_i(t) = t$), we assume each shift action to have the same unit cost, and we consider restricted types of shift actions. All these assumptions require some additional discussion.

The restrictions on the threshold functions and on the costs of shift actions seem to be very basic and, in fact, are even satisfied by the instances built in the proof of Theorem 2. The reason for assuming them is that, on the one hand, it seems beyond point to study instances more involved than those from Theorem 2, and, on the other hand, they interact with other restrictions, leading to tractable cases. But, they do have important consequences.

First, using identity threshold functions means that we model societies that are prone to propaganda. With identity threshold functions we cannot differentiate between voters that are more or less responsive to our actions. Second, assuming that every shift action has the same unit cost models settings where the costs of particular campaign actions are similar enough that small differences between them are irrelevant; the actual number of actions we choose to perform is a sufficiently good approximation of the real cost. This is true, for example, for the case of organizing meetings with voters, which often have comparable prices. It is also likely to be the case when shift actions model actions such as airing TV spots: Each spot has a similar cost to produce/broadcast. The greatest disadvantage of assuming unit costs is that we no longer can model mixed campaigns that use actions of several different types (meetings with voters, TV spots, web campaigns, etc.).

The restrictions on the types of allowed shift actions have even greater impact on the nature of campaigns that we study. We study the following classes of shift actions:

Unrestricted Shift Actions. Here we put no restrictions on the allowed shift actions; this models the most general (and, naturally, the least tractable) setting.

Bounded-Effect Shift Actions. Here we consider a parameter Γ and require that for each shift action $f = (f^{(1)}, \ldots, f^{(n)})$ it holds that for each j $(1 \leq j \leq n)$, we have $|f^{(j)}| \leq \Gamma$. This is still a very general setting, where we assume that each campaigning action has only a limited impact on each voter.

Unit-Effect Shift Actions. This is a class of bounded-effect shift actions for $\Gamma = 1$. For each given voter, applying a given shift action can either leave the preferred candidate p unaffected or it can shift p one position up or down.

Interval Shift Actions. This is a subclass of unit-effect shift actions that never affect voters negatively, and where for each shift action there is an interval of voters that are affected positively (the interval is with respect to the order of the voters in the input collection V). This class of shift actions models campaigns associated with a time window where certain voters can be reached, or campaigns that are local to given neighborhoods[3] (for example, that include putting up multiple posters, organizing meetings, etc.). We speak of 1^z-interval shift actions to mean interval shift actions where each shift action affects at most z voters.

Unit-Effect on Two Voters Shift Actions. This is a subclass of unit-effect shift actions that affect two voters at most. We focus on shift actions that affect both voters positively, denoted as $(+1, +1)$-shift actions, and that affect one voter positively and one voter negatively, denoted as $(+1, -1)$-shift actions. The reason for studying these families is not because they model particularly natural types of election campaigns, but rather to establish the limits of tractability for our problem. For example, we consider $(+1, -1)$-shift actions to understand how intractable are shift actions that have negative effects; $(+1, -1)$-shift actions are the simplest shift actions of this type that may be useful in the campaign (one would never deliberately use a shift action that only affects the preferred candidate negatively).

Figure 1 presents the difference between bounded-effect shift actions, unit-effect shift actions, unit-effect on two voters shift actions, and interval shift actions graphically. As we discuss in the next section, the type of allowed shift actions has a huge impact on the computational complexity of our problem.

3. In the neighborhood scenario, we take the simplified view that a society of the voters lives on a line. Of course, it would be more natural to take two-dimensional neighborhoods into account. We view this as an interesting direction for future research, but for the time being we consider as simple settings as possible. In the time window scenario, a natural ordering of the voters is the point of time when they cast their votes or can be affected by the campaign.

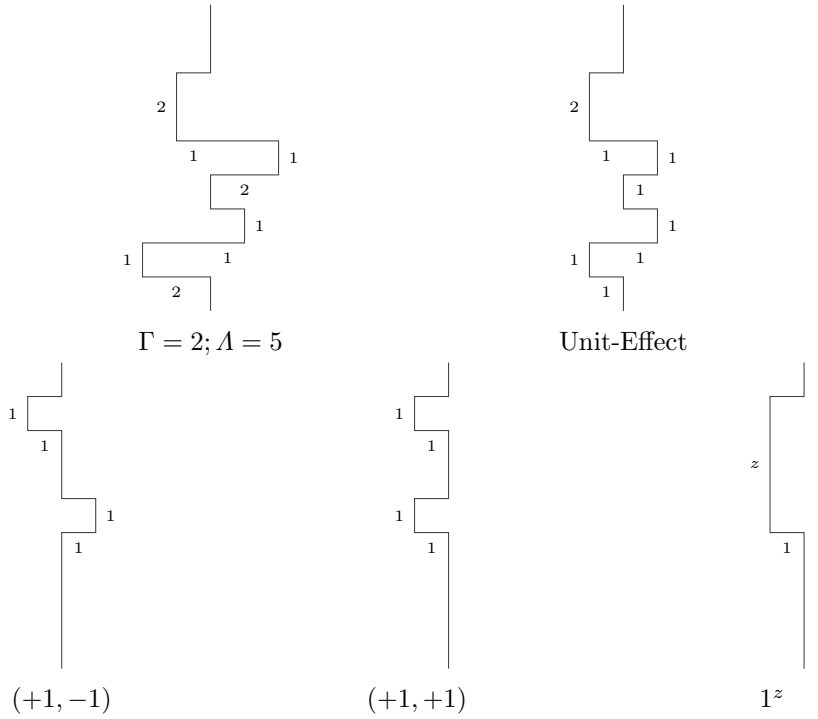


Figure 1: Restrictions on the shift actions. We visualize (from left to right, top to bottom): a shift action with maximum effect $\Gamma = 2$ of a single shift action and maximum number $\Lambda = 5$ of voters affected by a single shift action; a unit-effect shift action; a shift action with effect of $+1$ on one voter and effect of -1 on another voter "$(+1, -1)$"; a shift action with effect of $+1$ on two voters "$(+1, +1)$"; and a shift action with effect of $+1$ on an interval of size z "1^z". The intended interpretation is that voters are listed vertically, from top to bottom.

4. Overview of Results

We now provide a high-level overview of our results. It turns out that even with rather strong restrictions in place (that is, the restrictions defined in Section 3.4), Combinatorial Shift Bribery is computationally hard in most settings. What we present here is our quest for understanding the border between tractability and intractability of Combinatorial Shift Bribery. To this end, we employ the following techniques and ideas.

1. We seek both regular complexity results (NP-hardness results) and parameterized complexity results (FPT algorithms, W[1]-hardness and W[2]-hardness results, and XP algorithms).

2. We consider structural restrictions on the sets of available shift actions.

3. We seek approximation algorithms and inapproximability results (that is, approximation hardness results).

For our parameterized complexity results, we consider the following parameters: (a) the number n of the voters, (b) the number m of the candidates, (c) the budget B, (d) the maximum effect Γ of a single shift action, and (e) the maximum number Λ of voters affected by a single shift action.

All our discussions of (in)approximability of Combinatorial Shift Bribery regard the task of minimizing the cost of ensuring the preferred candidate's victory. This means that, for example, a 2-approximation algorithm has to decide if it is possible to ensure the preferred candidate's victory at all, and, if so, it has to output a successful set of shift actions with total cost at most twice as high as the optimal one.

We summarize our results in Table 1. These results show that Combinatorial Shift Bribery is highly intractable. Theorems 5, 6, and 7, show that the problem is computationally hard (in terms of NP-hardness, W[2]-hardness, and inapproximability even by FPT algorithms) for both the Plurality rule and the Borda rule, even for various very restricted forms of unit-effect shift actions, even for two candidates. This means that, in essence, the problem is hard for all natural voting rules, since for two candidates all natural voting rules boil down to the Plurality rule.

Further, Theorem 8 and Theorem 11 show that our problems are W[1]-hard even if we take the number of candidates and the budget as a joint parameter, even for extremely restricted shift actions. The problem remains hard (for the case of the Borda rule) when parameterized by the number of voters (Theorem 9). On the contrary, for the case of Plurality and parameterization by the number of voters we obtain tractability.

We obtain several approximability results. In essence, these results are possible only for the cases where shift actions do not have negative results. An intuitive reason for this fact is that when shift actions have negative effects, then it is computationally hard to check whether the preferred candidate can win even without any restrictions on the budget.

All our approximation algorithms are based on the results for the non-combinatorial variant of the problem, due to Elkind et al. (2009) and Elkind and Faliszewski (2010). Either we use the non-combinatorial algorithms directly, as subroutines in our algorithms, or we derive our results by plugging our Combinatorial Shift Bribery-specific blocks into the framework developed by Elkind et al. (2009) and Elkind and Faliszewski (2010).

5. Connection to Combinatorial Control

The study of combinatorial variants of problems modeling ways of affecting election results was initiated by Bulteau et al. (2015), who considered combinatorial control by adding voters (Combinatorial-CCAV) for the Plurality rule and for the Condorcet rule. It turns out that for the Plurality rule we can reduce the problem of (Combinatorial) CCAV to that of (Combinatorial) Shift Bribery. For the non-combinatorial variants of these problems this does not give much since both problems are easily seen to be polynomial-time solvable. However, there are strong hardness results for Plurality-Combinatorial-CCAV which we can transfer to the case of Plurality-Combinatorial Shift Bribery. Formally, Plurality-Combinatorial-CCAV is defined as follows (Bulteau et al., 2015).

Plurality-Combinatorial-CCAV
Input: A set C of candidates with a preferred candidate $p \in C$, a collection V

Table 1: Overview of our results. We show exact algorithms and approximation algorithms for Plurality-Combinatorial Shift Bribery and for Borda-Combinatorial Shift Bribery, for different restrictions on the shift actions (see Figure 1). Results marked by ▽ follow from the work of Elkind et al. (2009), by ♢ follow from the work of Bredereck et al. (2014a), by ♠ follow from the work of Elkind and Faliszewski (2010), and by ♡ follow from the work of Bredereck et al. (2016). Note that all of the variants are in XP when parameterized by the budget B (Observation 1).

shift actions	rule	exact complexity	approximability
regular Shift Bribery (convex prices)	Plurality	poly.-time solvable (▽)	—
	Borda	NP-complete(▽), in FPT for B (♢), W[1]-hard for n (♡)	2-approximable in poly. time (♠,▽), FPT-approximation scheme for n (♢)
unit effect	Both	W[2]-h for B even if $m = 2$ (Thm. 5), XP for n (Prop. 12)	inapproximable even in FPT-time for B even if $m = 2$ (Thm. 6)
$(+1, -1)$	Plurality	FPT for n (Thm. 13)	—
	Borda	W[1]-hard for n (Thm. 9)	inapproximable even in FPT-time for n (Cor. 10)
	Both	NP-h even if $m = 2$ (Thm. 7), W[1]-h for B and m combined (Thm. 8)	inapproximable even if $m = 2$ (Thm. 7)
$(+1, +1)$	Plurality	FPT for n (Thm 13)	—
	Both	W[1]-h for B and m combined (Thm. 8)	2-approximable in poly. time (Thm. 15)
1^z-intervals	Plurality	FPT for n (Thm. 13)	z-approximable in poly. time (Thm. 14)
	Borda	—	$2z$-approximable in poly. time (Thm. 14)
	Both	W[1]-h for B (Thm. 11)	2-approximable in m^z time (Thm. 16)

of registered voters (having preference orders over C), a collection W of unregistered voters (having preference orders over C), a bundling function $\kappa\colon W \to 2^W$ (for each $w \in W$ it holds that $w \in \kappa(w)$), and a budget k.
Question: Is there a collection $W' \subseteq W$ of at most k voters such that p is a winner of the modified election $(C, V \cup \bigcup_{w' \in W'} \kappa(w'))$?

Intuitively, for each unregistered voter $w \in W$, we have her bundle, $\kappa(w)$ (given explicitly in the input), such that when we add w to the election (for example, by somehow convincing her to vote), all the voters in her bundle also join the election (for example, people choose to vote under an influence of a friend).

Theorem 3. *Plurality-*Combinatorial-CCAV *is polynomial-time many-one reducible to Plurality-*Combinatorial Shift Bribery*. For an instance of Plurality-*Combinatorial-CCAV *with m candidates, the reduction outputs an instance of Plurality-*Combinatorial Shift Bribery *with $m + 1$ candidates.*

Proof. Consider an input instance of Plurality-Combinatorial-CCAV with candidate set C, collection of registered voters V, collection of unregistered voters W, bundling function κ, preferred candidate $p \in C$, and limit k on the number of voters that we can add. We form an instance of Plurality-Combinatorial Shift Bribery, as follows.

We form a candidate set $C' = C \cup \{d\}$, where d is some new candidate. We form the set of voters V' in the following way.

1. For each voter $v \in V$, we include v in V', with the preference orders extended to rank d last.

2. For each voter $w \in W$ that ranks p first, we include in V' two voters, x_w, with preference order of the form $d \succ p \succ \cdots$, and x'_w, with preference order of the form $p \succ d \succ \cdots$.

3. For each voter $w \in W$ that ranks some candidate $c \in C \setminus \{p\}$ first, we include in V' voter x_w with preference order $p \succ c \succ \cdots$, and voter x'_w with preference order $d \succ p \succ \cdots$.

4. We include $4|W||C|$ voters in V' with preference orders such that we will achieve the following effects: (a) for each $c \in C$ with score $s(c)$ in election (C, V), c is ranked first by $4|W| + s(c)$ voters in V', and (b) d is ranked first by exactly $2|W|$ voters in V'. To achieve these effects, for each $c \in C \setminus \{p\}$ we include $4|W|$ voters that rank c first, we include $3|W|$ voters that rank p first, and we include $|W|$ voters that rank d first.

For each voter $w \in W$, we introduce a shift action f_w with the following effects: for each $w' \in \kappa(w)$, if w' ranks p first then f_w has effect 1 on $x_{w'}$ (but not on $x'_{w'}$) and if w' ranks some candidate in $C \setminus \{p\}$ first, then f_w has effect -1 on $x_{w'}$ and effect $+1$ on $x'_{w'}$ (all other entries are zeros). This finishes the construction. We provide the proof of correctness after the following example of the reduction.

Example 2. Consider the following input to Plurality-Combinatorial-CCAV, where the preferred candidate is p and the budget k is 1.

registered voters	unregistered voters	bundling function
$v_1 : p \succ a$	$w_1 : p \succ a$	$\kappa(w_1) = \{w_1, w_3\}$
$v_2 : a \succ p$	$w_2 : a \succ p$	$\kappa(w_2) = \{w_2\}$
$v_3 : a \succ p$	$w_3 : p \succ a$	$\kappa(w_3) = \{w_2, w_3\}$

We construct the following input to Plurality-Combinatorial Shift Bribery; notice that the number of entries in each shift action is 33.

election	shift actions		
$v_1 : p \succ a \succ d$	$\begin{pmatrix}0\\0\\0\\1\\0\\0\\0\\0\\1\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\\0\\0\\-1\\1\\0\\0\\0\\0\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\\0\\0\\-1\\1\\1\\0\\0\\0\\0\\0\end{pmatrix}$
$v_2 : a \succ p \succ d$			
$v_3 : a \succ p \succ d$			
$x_{w_1} : d \succ p \succ a$			
$x'_{w_1} : p \succ d \succ a$			
$x_{w_2} : p \succ a \succ d$			
$x'_{w_2} : d \succ p \succ a$			
$x_{w_3} : d \succ p \succ a$			
$x'_{w_3} : p \succ d \succ a$			
12 dummies : $a \succ \cdots$			
9 dummies : $p \succ \cdots$			
3 dummies : $d \succ \cdots$			
	f_{w_1}	f_{w_2}	f_{w_3}

Note that adding voter w_1 to the input election for Plurality-Combinatorial-CCAV results in p being a winner of the election. Correspondingly, applying shift action f_{w_1} results in p being a winner of the input election for Plurality-Combinatorial Shift Bribery. $\triangle$

To see the correctness of our construction, note that applying a shift action corresponding to a bundle of a voter $w \in W$ has the same effect on the differences between the scores of the candidates in C as adding the bundle $\kappa(w)$ has in the original control instance. More specifically, disregarding the score of d for now, we have the following. For each $w' \in \kappa(w)$ which ranks p first, we have an increase of the score of p by one, while for each $w' \in \kappa(w)$ which ranks some candidate $c \in C \setminus \{p\}$ first, we have an increase of the score of c by one. Further, the score of candidate d can never grow beyond $4|W|$ in our Plurality-Combinatorial Shift Bribery instance and the score of p can never fall below $4|W|$. Therefore, d can never prevent p from being a winner.

Thus, the reduction is correct. Furthermore, the reduction can be computed in polynomial time and it outputs a Plurality-Combinatorial Shift Bribery instance with one candidate more than the input Plurality-Combinatorial-CCAV instance. We also observe that the output instance uses unit-effect shift actions that affect at most twice as many voters as the largest bundle in the input instance. $\square$

Based on the proof of Theorem 3 and results of Bulteau et al. (2015), we obtain the following result.

Corollary 4. *Plurality-*Combinatorial Shift Bribery *is $W[2]$-hard with respect to the budget B even if $m = 3$, it is $W[1]$-hard with respect to B even for shift actions with unit effect on up to 6 voters, and it is* NP*-hard even for shift actions with unit effects on up to 4 voters.*

Proof. The result follows by applying the reduction from the proof of Theorem 3 to the Plurality-Combinatorial-CCAV instances produced in the reductions from Theorems 2, 1, and 4 of Bulteau et al. (2015), respectively. □

6. Hardness Results

The results from the previous section show that we are bound to hit hard instances for Combinatorial Shift Bribery even in very restricted setting. In this section we explore how restrictive these hard settings are. Our results are organized by the type of shift actions allowed.

6.1 Results for General Unit-Effect Shift Actions

We start by considering unit-effect shift actions. If the allowed effects are positive only, then we obtain NP-hardness and W[2]-hardness when parameterizing by the budget B. If we allow also negative unit-effects, then the problem gets even harder and we go beyond any hope for an approximation algorithm, even if the approximation algorithm were allowed to run in FPT time when parameterizing by the budget B. Quite strikingly, these results hold even if we only have two candidates.

Theorem 5. *For both the Plurality rule and the Borda rule,* Combinatorial Shift Bribery *is* NP*-hard and* W[2]*-hard for the parameter budget B, even for two candidates and even if each shift action has effects of either $+1$ or 0 on each voter.*

Proof. We provide a parameterized reduction from Set Cover (recall Section 2.3). Let $(\mathcal{S}, X, h)$ be an instance of Set Cover, where $\mathcal{S} = \{S_1, \ldots, S_m\}$ is a family of subsets over the universe $X = \{x_1, \ldots, x_n\}$, and h is the number of sets that we can use to cover X. We construct an instance of Plurality-Combinatorial Shift Bribery with two candidates. Note that, since the Borda rule and the Plurality rule coincide on elections with two candidates, our hardness result transfers to Borda-Combinatorial Shift Bribery.

The construction is as follows. We have p and d as the only candidates. For each element $x_i \in X$ create an *element voter* v_i with preference order $d \succ p$. Create another set of n *dummy voters* all with preference order $d \succ p$. The set F of shift actions contains for each set $S_j \in \mathcal{S}$ a function f_j having an effect of $+1$ on the element voters corresponding to the elements of the set (that is, $f_j[i] = 1$ if $x_i \in S_j$ and $f_j[i] = 0$ otherwise). Finally, set $B := h$. This finishes the construction. Clearly, the reduction can be computed in polynomial time. Consider the following example of applying the reduction.

Example 3. Let the input to Set Cover be such that $X = \{x_1, x_2, x_3, x_4, x_5\}$ and $S = \{S_1, S_2, S_3\}$, with $S_1 = \{1, 2, 5\}$, $S_2 = \{2, 3\}$, $S_3 = \{3, 4\}$, and $h = 2$. We construct the following input for Plurality-Combinatorial Shift Bribery.

election	shift actions		
$v_1 : d \succ p$	1	0	0
$v_2 : d \succ p$	1	1	0
$v_3 : d \succ p$	0	1	1
$v_4 : d \succ p$	0	0	1
$v_5 : d \succ p$	1	0	0
5 dummies : $d \succ p$	0	0	0
	f_1	f_2	f_3

Note that $\{S_1, S_3\}$ is a set cover, and, analogously, choosing f_1 and f_3 results in p being a winner of the election. △

It remains to show that there is a set cover of size h if and only if there is a successful set of shift actions of size h.

For the "if" part, assume that there is a set cover $\mathcal{S}'$ of size at most h. Then, applying $F' = \{f_j \mid S_j \in \mathcal{S}'\}$ makes p win the election: Since $\mathcal{S}'$ is a set cover, p will be the preferred candidate of all n element voters and, hence, a winner of the election.

For the "only if" part, assume that there is a set of shift actions $F' \subseteq F$ of size at most h whose application makes p win the election. Then, p must be the preferred candidate of all element voters in the bribed election because no shift action has effect on any dummy voter. Since there are n element voters and n dummy voters, $\mathcal{S}' := \{S_j \mid f_j \in F'\}$ is a set cover. Finally, since $B = h$, S' is of size at most h. □

Allowing also negative (but unit) effects on the voters, we can adapt our reduction from Theorem 5 to show a strong inapproximability result. The inapproximability result follows since in the corresponding reduction, for yes-instances, the only correct solutions use the exact given budget.

Theorem 6. *Unless* W[2] = FPT, Combinatorial Shift Bribery *is inapproximable (in* FPT *time for the parameter B) for both the Plurality rule and the Borda rule, even for two candidates and unit-effect shift actions.*

Proof. We modify the reduction from Theorem 5 to show our inapproximability result. Let $(\mathcal{S}, X, h)$ be a Set Cover instance where $\mathcal{S} = \{S_1, \ldots, S_m\}$ and $X = \{x_1, \ldots, x_n\}$. Without loss of generality, we assume that $|\mathcal{S}| > h$. We construct an instance of Plurality-Combinatorial Shift Bribery with two candidates as follows. (Since we have two candidates only, the proof applies to the case of Borda-Combinatorial Shift Bribery as well.)

For each element $x_i \in X$, create $|\mathcal{S}|$ *element voters* $v_i^1, \ldots, v_i^{|\mathcal{S}|}$, each with preference order $d \succ p$, and for each set $S_j \in \mathcal{S}$ create a *set voter* v_j^0 with preference order $p \succ d$. Create $|\mathcal{S}| \cdot |X| + |\mathcal{S}| - 2h$ *dummy voters*, each with preference order $d \succ p$. The set F of shift actions contains, for each set S_j, a shift action f_j having an effect of 1 on each element voter corresponding to an element of the set and an effect of -1 on the set voter corresponding to the set. Finally, set $B := h$. This completes the construction, which is clearly computable in polynomial time.

Next, we show that there is a successful set of shift actions of size h if and only if there is a set cover of size h.

For the "if" part, assume that there is a set cover $\mathcal{S}'$ of size at most h. Then, $F' = \{f_j \mid S_j \in \mathcal{S}'\}$ is a successful set of shift actions: since $\mathcal{S}'$ is a set cover, p will be the preferred candidate of all $|\mathcal{S}| \cdot |X|$ element voters and also the preferred candidate for at least $|\mathcal{S}| - h$ set voters (corresponding to the sets not from the set cover). Moreover, d will be the preferred candidate for all $|\mathcal{S}| \cdot |X| + |\mathcal{S}| - 2h$ dummy voters and also the preferred candidate for at most h set voters (corresponding to the sets from the set cover). Hence, either p wins or p and d tie as winners.

For the "only if" part, assume that there is a successful set of shift actions $F' \subseteq F$ of size at most h. Then, p must be the preferred candidate for all element voters in the bribed election: If there were an element voter with $d \succ p$, then there would be at least $|\mathcal{S}| - 1$ further element voters with $d \succ p$ (the element voters corresponding to the same element). Thus there would be in total at most $|\mathcal{S}|(|X| - 1)$ element voters and $|\mathcal{S}|$ set voters that prefer p, but at least $|\mathcal{S}| \cdot |X| + |\mathcal{S}| - 2h$ dummy voters and $|\mathcal{S}|$ element voters that prefer d. Since we assumed $|\mathcal{S}| > h$, this would mean that p is not a winner. Thus, it must be that $\mathcal{S}' := \{S_j \mid f_j \in F'\}$ is a set cover, and, due to the budget constraint, it follows that $|\mathcal{S}'| \leq h$.

Finally, we show that Plurality-Combinatorial Shift Bribery is inapproximable even in FPT time when parameterized by the budget. Assume, for the sake of a contradiction, that a successful set of shift actions $F' \subseteq F$ with $|F'| > B$ exists. Then, in the bribed election, at least $|\mathcal{S}| \cdot |X| + |\mathcal{S}| - 2h$ dummy voters and also $|F'| \geq h + 1$ set voters prefer d, but at most $|\mathcal{S}| \cdot |X|$ element voters and at most $|\mathcal{S}| - (h + 1)$ set voters prefer p. Thus, d is the unique winner. Hence, *any* successful bribery action must be optimal with respect to the budget and any FPT-algorithm for Plurality-Combinatorial Shift Bribery (parameterized by the budget) would solve the W[2]-hard problem Set Cover (parameterized by the solution size) in FPT time; a contradiction to the assumption that FPT $\neq$ W[2]. □

6.2 Results for Shift Actions with Unit Effect on Two Voters

In the previous section we did not limit the number of voters affected by each shift action. Now we focus on the case where each unit-effect shift action can affect at most two voters. First we show that Combinatorial Shift Bribery remains NP-hard and hard to approximate for $(+1, -1)$-shift actions. Then we provide parameterized hardness results for both $(+1, -1)$ and $(+1, +1)$-shift actions. The proof is relatively similar to the one for Theorem 6 and so we defer it to Appendix A.

Theorem 7. *Unless* P = NP, Combinatorial Shift Bribery *is inapproximable (in polynomial time) for both the Plurality rule and the Borda rule, even for two candidates and* $(+1, -1)$*-shift actions.*

As opposed to Theorem 6, the above result does not yield W[2]-hardness for the parameter budget B. This is because our proof uses a reduction from Set Cover in which the value of the budget is the size of the universe set X. If we insist on parameterized hardness for unit effects on two voters, then we have to accept larger sets of candidates. However, this increase is not too large: below we show W[1]-hardness of Combinatorial Shift Bribery jointly parameterized by the budget and the number of candidates.

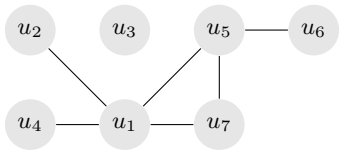
Theorem 8. *For both the Plurality rule and the Borda rule,* Combinatorial Shift Bribery *is* W[1]*-hard for the combined parameter* (m, B)*, even if we either only have* $(+1, -1)$*-shift actions or only have* $(+1, +1)$*-shift actions.*

Proof. We have four cases to consider. We begin with the Plurality rule and $(+1, +1)$-shift actions.

The Plurality Rule with $(+1, +1)$-Shift Actions. We describe a parameterized reduction from the W[1]-hard Clique problem, parameterized by the solution size, to Plurality-Combinatorial Shift Bribery with $(+1, +1)$-shift actions, parameterized by (m, B).

Let (G, h) be an instance of Clique with $V(G) = \{u_1, \ldots, u_{n'}\}$ and $E(G) = \{e_1, \ldots, e_{m'}\}$. We create the following instance of Plurality-Combinatorial Shift Bribery. The set of candidates is $\{p\} \cup D$, where $D = \{d_1, \ldots, d_{h-1}\}$. For each vertex $u_i \in V(G)$, we create a *vertex voter* v_i with preference order $\overrightarrow{D} \succ p$. Moreover, we create $n' - 2h$ *dummy voters* with preference order $p \succ \overrightarrow{D}$ each. For each edge $\{u_i, u_j\} \in E(G)$, we create a shift action $f_{\{u_i,u_j\}}$ with effect 1 on the vertex voters v_i and v_j, and effect 0 on all other voters. Finally, we set the budget to $B := \binom{h}{2}$. This completes the construction, which is computable in polynomial time. Consider the following example.

Example 4. We have the following graph, where we are looking for a clique of size $h = 3$.



We construct the following input for Plurality-Combinatorial Shift Bribery.

election	**shift actions**					
$v_1 : d_1 \succ d_2 \succ p$	1	1	1	1	0	0
$v_2 : d_1 \succ d_2 \succ p$	1	0	0	0	0	0
$v_3 : d_1 \succ d_2 \succ p$	0	0	0	0	0	0
$v_4 : d_1 \succ d_2 \succ p$	0	1	0	0	0	0
$v_5 : d_1 \succ d_2 \succ p$	0	0	1	0	1	1
$v_6 : d_1 \succ d_2 \succ p$	0	0	0	0	1	0
$v_7 : d_1 \succ d_2 \succ p$	0	0	0	1	0	1
1 dummy : $p \succ d_1 \succ d_2$	0	0	0	0	0	0
	f_{u_1,u_2}	f_{u_1,u_4}	f_{u_1,u_5}	f_{u_1,u_7}	f_{u_5,u_6}	f_{u_5,u_7}

Note that (v_1, v_5, v_7) form a clique of size 3 in the input graph for Clique, and, accordingly, applying the set of shift actions $\{f_{u_1,u_5}, f_{u_1,u_7}, f_{u_5,u_7}\}$ results in p being the winner of the election for Plurality-Combinatorial Shift Bribery. △

Without loss of generality, assume that d_1 is ranked first in the (arbitrary but fixed) order $\overrightarrow{D}$. Observe that we have n' vertex voters and h dummy voters which rank d_1 first. We also have $n' - h$ dummy voters which rank p first. Hence, to make p win the election, one needs h additional voters to rank p first (and, in effect, not rank d_1 first).

It remains to show that our constructed instance contains a successful set of shift actions F' of size h if and only if (G, h) contains a clique of size h.

For the "if" part, let $H \subseteq V(G)$ be a set of h vertices forming a clique and let $E' \subseteq E(G)$ be the set of edges between the vertices from H. Then, observe that $F' = \{f_{\{u_i,u_j\}} \mid \{u_i, u_j\} \in E'\}$ is a successful set of shift actions: For each vertex voter v_i corresponding to a clique vertex $u_i \in H$, candidate p is shifted $h - 1$ positions forward. This means that, in total, we have that h vertex voters rank p first and p ties as a winner of the election.

For the "only if" part, let F' be a successful set of shift actions. Since dummy voters are not affected by any shift action, it follows that in order to make p a winner of the election, p must be shifted to the top position in at least h vertex voters. That is, in total, p must be shifted $h \cdot (h - 1)$ positions forward. Since F' is of size at most $B = \binom{h}{2} = h \cdot (h - 1)/2$ and each shift action affects only two vertex voters, F' must be of size exactly $\binom{h}{2}$ affecting exactly h vertex voters. By construction, this implies that there are $\binom{h}{2}$ edges in G incident to exactly h different vertices which is only possible if these h vertices form a clique. This finishes the proof for the Plurality rule with $(+1, +1)$-shift actions.

The remaining cases of the proof are quite similar (although, technically, more involved) and we present them in Appendix B. □

It is quite natural to consider Combinatorial Shift Bribery also from a different perspective. Instead of asking what happens for a small number of candidates, we might ask about the complexity of Combinatorial Shift Bribery for a small number of voters (see, for example, Brandt, Harrenstein, Kardel, & Seedig, 2013; Chen et al., 2015, for some motivation as to why looking at elections with few voters is interesting). In this case we obtain hardness only for the Borda rule. Indeed, later we will show that Plurality-Combinatorial Shift Bribery is in FPT for the parameter number of voters. The proof of the next theorem is quite involved and is available in Appendix C.

Theorem 9. *Borda-*Combinatorial Shift Bribery *is* W[1]*-hard with respect to the number n of voters, even for $(+1, -1)$-shift actions and no budget constraints.*

In the proof of Theorem 9 we reduce from the Strongly Regular Multicolored Clique problem, and, importantly, we do not impose any budget constraints. Thus, it follows that any approximation algorithm for Borda-Combinatorial Shift Bribery (running in FPT time when parameterized by the number of voters) would yield an FPT algorithm for Strongly Regular Multicolored Clique when parameterized by the solution size. In effect, we have the following corollary.

Corollary 10. *Unless* W[1] = FPT*, Borda-*Combinatorial Shift Bribery *is inapproximable even in* FPT*-time for the parameter n, even for $(+1, -1)$-shift actions.*

The results from Theorem 9 and Corollary 10 compare very interestingly to those for the non-combinatorial variant of Borda-Shift Bribery. Until very recently, the complexity of Borda-Shift Bribery parameterized by the number of voters was unknown. Eventually

(in a different paper, and after submitting this one for journal publication) we have shown that the problem is W[1]-hard (Bredereck et al., 2016), through a far simpler proof than the one used here. Nonetheless, Theorem 9 and Corollary 10 still carry significant value. Earlier, Bredereck et al. (2014a) have shown that there is an FPT approximation scheme for Borda-Shift Bribery parameterized by the number of voters, and Corollary 10 shows that this result does not generalize to the combinatorial setting.

6.3 Results for Interval Shift Actions

We conclude the discussion of hardness results by considering Combinatorial Shift Bribery with interval shift actions. In the previous section we allowed shift actions to have non-zero effects on two voters each, but these two voters could have been chosen arbitrarily. Now we show a hardness result for the case where we can positively affect multiple voters, but these voters have to form a consecutive interval in the input election.

Theorem 11. *For both the Plurality rule and the Borda rule,* Combinatorial Shift Bribery *is* NP*-hard even for interval shift actions.*

Proof. We consider the Plurality rule first and give a many-one reduction from the following variant of the strongly NP-hard Numerical Matching with Target Sums problem.

> Numerical Matching with Target Sums
> **Input:** Three sets of integers $A = \{a_1, \ldots, a_t\}$, $B = \{b_1, \ldots, b_t\}$, and $X = \{x_1, \ldots, x_t\}$, where (1) the numbers are encoded in unary, (2) all the $3t$ numbers are distinct, and (3) no two numbers that are both from A or both from B sum up to any number in X.
> **Question:** Can the elements of A and B be paired so that for each $i \in [t]$ the sum of the ith pair is exactly x_i?

The standard variant of the problem, as presented in the classic text of Garey and Johnson (1979), does not have any restrictions on the integers in sets A, B, and X. We can assume that the numbers are encoded in unary because the problem is strongly NP-hard. Further, Hulett, Will, and Woeginger (2008) have shown that the problem remains NP-hard for the case where all the $3t$ integers are distinct. Finally, to see that the third restriction does not change the complexity of the problem it suffices to consider the following transformation: Given an instance (A, B, X) of Numerical Matching with Target Sums, we add $2 \cdot \max(A \cup B \cup X) + 1$ to each integer in B and X. This produces an equivalent instance where no two numbers, both from A or both from B, sum up to any number in X.

The Plurality Rule. Let (A, B, X) be an instance of Numerical Matching with Target Sums and let y denote the largest integer in $A \cup B \cup X$. We create an instance of Plurality-Combinatorial Shift Bribery as follows. The set of candidates is:

$$C := \{p, d, c_1^a, \ldots, c_t^a, c_1^b, \ldots, c_t^b, c_1^x, \ldots, c_t^x\}.$$

We create the following voters.

1. For each pair of integers $a_i \in A$ and $x_\ell \in X$, we introduce:

   (a) One voter with preference order

   $$c_i^a \succ p \succ \overrightarrow{C \setminus \{p, c_i^a\}},$$

   (b) a_i voters each with preference order

   $$c_\ell^x \succ p \succ \overrightarrow{C \setminus \{p, c_\ell^x\}},$$

   (c) $2y - (a_i + 1)$ voters each with preference order

   $$d \succ p \succ \overrightarrow{C \setminus \{p, d\}}.$$

   These voters are called the (a_i, x_ℓ)-voters and there are exactly $2y$ of them. For each pair (a_i, x_ℓ), we construct a shift action $f_{a_i}^{x_\ell}$ with effect 1 on exactly the set of (a_i, x_ℓ) voters.

2. For each pair of integers $b_j \in B$ and $x_\ell \in X$, we introduce:

   (a) One voter with preference order

   $$c_j^b \succ p \succ \overrightarrow{C \setminus \{p, c_j^b\}},$$

   (b) b_j voters each with preference order

   $$c_\ell^x \succ p \succ \overrightarrow{C \setminus \{p, c_\ell^x\}},$$

   (c) $2y - (b_j + 1)$ voters each with preference order

   $$d \succ p \succ \overrightarrow{C \setminus \{p, d\}}.$$

   These voters are called the (b_j, x_ℓ)-voters and there are exactly $2y$ of them. For each pair (b_j, x_ℓ), we construct a shift action $f_{b_j}^{x_\ell}$ with effect 1 on exactly the set of (b_j, x_ℓ) voters.

3. Let $q := 4ty$. We create sufficiently many *dummy voters* to ensure that, altogether, the candidates have the following scores:

   (a) p has q points,

   (b) for each i, c_i^a and c_i^b have $q + 4ty + 1$ points each, and

   (c) for each $\ell \in [t]$, c_ℓ^x has $q + 4ty + x_\ell$ points.

   No shift action affects any of the dummy voters.

Finally, we set the budget $B := 2t$. This completes the reduction. It is easy to see that it is computable in polynomial time (because all the numbers are encoded in unary) and that we can order the voters so that each shift action effects on a consecutive interval of $z := 2y$ voters.

It remains to show that our constructed instance of Plurality-Combinatorial Shift Bribery contains a successful set F' of shift actions of size at most $2t$ if and only if (A, B, X) is a "yes"-instance of Numerical Matching with Target Sums.

For the "if" part, let $S := \{(a_{i_1}, b_{j_1}), \ldots, (a_{i_t}, b_{j_t})\}$ be a solution for Numerical Matching with Target Sums, that is, a set of integer pairs such that each integer from $A \cup B$ occurs exactly once in S and such that $a_{i_\ell} + b_{j_\ell} = x_\ell$ holds for each $\ell \in [t]$. Observe that $F' := \{f_{a_{i_\ell}}^{x_\ell}, f_{b_{j_\ell}}^{x_\ell} \mid (a_{i_\ell}, b_{j_\ell}) \in S\}$ is a successful set of shift actions. Since each integer from $A \cup B$ occurs exactly once in (some pair of) S, each candidate c_i^a and each candidate c_j^b loses one point. Since $a_{i_\ell} + b_{j_\ell} = x_\ell$ for each $\ell \in [t]$, each candidate c_ℓ^x loses x_ℓ points. By construction, p gains $4ty$ points by any set of shift actions of size $2t$. Thus, p wins the election.

For the "only if" part, let F' be a successful set of shift actions of size $2t$ (if there were a successful action of smaller size we could extend it to size $2t$ because our shift actions do not have negative effects). After applying shift actions from F', p gains $4ty$ points. If this is to make p a winner of the election, each candidate c_i^a and each candidate c_j^b needs to lose one point, and each candidate c_ℓ^x needs to lose x_ℓ points. Thus, for each $a_i \in A$ there is exactly one $f_{a_i}^{x_{\ell_i}} \in F'$ and for each $b_j \in B$ there is exactly one $f_{b_j}^{x_{\ell_i}} \in F'$. Since all the integers in $A \cup B \cup X$ are distinct and no two integers both from A or both from B sum up to any integer from X, for each $x_\ell \in X$ there is at least one shift action $f_{a_{i_\ell}}^{x_\ell}$ with effect on a_{i_ℓ} voters who prefer c_l^x, and one shift action $f_{b_{j_\ell}}^{x_\ell}$ with effect on b_{j_ℓ} voters who prefer c_ℓ^x. Since there are t candidates c_ℓ^x and $|F'| = 2t$, it follows that there are exactly two shift actions with effect on some voters preferring c_ℓ^x. Since c_ℓ^x has to lose at least x_ℓ points, it holds that $a_{i_\ell} + b_{j_\ell} \geq x_\ell$. In fact, by the pigeonhole principle, it holds that $a_{i_\ell} + b_{j_\ell} \geq x_\ell$. Hence, if there is a successful set of $2t$ shift actions, then there is a solution for our Numerical Matching with Target Sums instance.

The Borda Rule. For the Borda rule, almost the same reduction works. Specifically, there still exists some integer q for which the set of requirements which were required in the proof for the Plurality rule will now hold for the Borda rule (with respect to a different q). Importantly, since p is in the second position in the preference profiles of all of the voters, it holds that the score differences, when applying some shift actions, are similar for the Plurality rule and the Borda rule. Thus, the proof of correctness for the Plurality rule transfers to the Borda rule. □

Throughout this section we have shown a number of hardness results under more and more restrictive assumptions regarding the available shift actions. In the following sections we seek positive algorithmic results.

7. Exact Algorithms

In spite of the pessimism looming from the previous section, in this section we show two exact FPT and XP algorithms for $\mathcal{R}$-Combinatorial Shift Bribery. Then, in Section 8, we present several efficient approximation algorithms.

We begin by observing that $\mathcal{R}$-Combinatorial Shift Bribery can be solved in polynomial time, provided that we assume the budget B to be a constant. The reason is that we need to choose at most B shift actions out of all available ones, but the number of shift actions available is upper-bounded by the input size.

Observation 1. *Both Plurality-*Combinatorial Shift Bribery *and Borda-*Combinatorial Shift Bribery *are in* XP *when parameterized by the budget B.*

If we restrict the instances to contain only bounded-effect shift actions, then we can show that $\mathcal{R}$-Combinatorial Shift Bribery can be solved in polynomial time, provided that the number n of the voters is treated as a constant.

Proposition 12. *If the maximum effect of every shift action is upper-bounded by some universal constant, then both Plurality-*Combinatorial Shift Bribery *and Borda-*Combinatorial Shift Bribery *are in* XP *when parameterized by the number n of the voters.*

Proof. Let Γ be the value bounding, component-wise, the effect of each shift action. First, observe that there are at most $(2\Gamma + 1)^n$ types of different shift actions. Second, observe that once one knows the budget spent on each type of shift actions, one can easily check whether a corresponding set of shift actions makes p a winner of the election. Thus we use the following algorithm: We try all possibilities of distributing the budget B among the at most $(2\Gamma + 1)^n$ types of shift actions and check whether one of them makes p a winner. If so, we accept. Otherwise we reject. □

Proposition 12 holds even if each shift action comes at an individual cost and if each voter has an individual threshold function, because we can, given some budget, always select the cheapest set of shift actions of a given type. Further, by expressing our problem as an integer linear program (ILP) and by using a famous result of Lenstra (1983), for the Plurality rule we can strengthen the above XP-membership to FPT-membership.

Theorem 13. *For bounded-effect shift actions (where we treat the bound as a universal constant), Plurality-*Combinatorial Shift Bribery *is in* FPT *when parameterized by the number n of the voters.*

Proof. Given an instance of Plurality-Combinatorial Shift Bribery with n voters, our algorithm proceeds as follows. First, we guess a subset of the voters for whom we will guarantee that p is ranked first (there are 2^n guesses to try). For each guessed set of voters, we test whether p would be a winner of the election if p were shifted to the top position by the guessed voters and was not ranked first by the remaining voters. For each guessed subset V' of voters for which this test is positive, we check whether it is possible to ensure (by applying shift actions whose cost does not exceed the budget) that the voters from V' rank p first. We do so as follows.

Let Γ be the universal constant bounding, component-wise, the effect of each shift action. Observe that there are at most $(2\Gamma+1)^n$ types of different shift actions. For each shift action

type z, we introduce a variable x_z denoting the number of times a shift action of type z is present in the solution. For each voter v_i, denote by $s_{v_i}(p)$ the position of p in the original preference order of v_i. For each voter $v_i \in V'$, we add the following constraint:

$$\sum_{\gamma \in [-\Gamma, \Gamma]} \left(\gamma \sum_{\{z : f_z \text{ has an effect of } \gamma \text{ on } v_i\}} x_z \right) \geq s_{v_i}(p).$$

This ensures that p is indeed shifted to the top position in v_i's preference list. We add the budget constraint:

$$\sum x_z \leq B,$$

ensuring that the solution respects the budget. Finally, for each shift action type z we add a constraint ensuring that we use at most as many shift actions of type z as there are available in the input. This finishes the description of the ILP. By a result of Lenstra (1983), we can solve this ILP in FPT time, because we have at most $(2\Gamma + 1)^n$ integer variables. □

Roughly speaking, Theorem 13 is the reason why Theorem 9 does not apply to the Plurality rule. In this setting, Plurality-Combinatorial Shift Bribery is tractable. Note that Theorem 13 applies to the case where each shift action has the same unit cost, i.e., the case on which we focus in this paper. Nonetheless, we believe that it is possible to lift Theorem 13 to the case where each shift action has its individual cost, by applying ideas of Bredereck, Faliszewski, Niedermeier, Skowron, and Talmon (2015a).

8. Approximation Algorithms

We now explore the possibility of finding approximate solutions for Combinatorial Shift Bribery. We focus on approximating the cost of shift actions necessary to ensure p's victory (for example, a 2-approximate algorithm finds a solution that ensures p's victory whenever it is possible, and uses at most twice as many shift actions as necessary). By Theorems 6 and 7, we know that we cannot hope to find approximate algorithms for the cases of Combinatorial Shift Bribery where the shift actions can have negative effects. Thus, in this section, we focus on unit-effect shift actions with only positive effects. This also simplifies our situation in that we can always check if it is possible to ensure p's victory: It suffices to apply all the available shift actions and check if p is a winner (indeed, not being able to perform such a check is at the heart of our inapproximability results from Section 6).

All our approximation algorithms proceed either by directly invoking the algorithms for the non-combinatorial variant of Shift Bribery of Elkind et al. (2009) and Elkind and Faliszewski (2010), or by plugging our algorithms into their framework. We start with the former approach and then describe the latter.

Theorem 14. *If each shift action has effects of either* 0 *or* 1 *on each voter, then Plurality-*Combinatorial Shift Bribery *can be Λ-approximated in polynomial-time and Borda-*Combinatorial Shift Bribery *can be 2Λ-approximated in polynomial time, where Λ denotes the maximum number of voters affected by a shift action.*

Proof. The general idea of these approximation algorithms is to split each shift action that affects some $\Lambda' \leq \Lambda$ voters into Λ' shift actions, each affecting a single voter only. In effect

we construct a non-combinatorial instance of Shift Bribery that we solve exactly, for the case of Plurality rule, or 2-approximately, for the case of the Borda rule.

Specifically, our construction goes as follows. Let $\lambda(i)$ denote the number of shift actions affecting voter i. Given an instance of Combinatorial Shift Bribery, we form an instance of Shift Bribery that is identical, except that instead of having shift actions, we have price functions for the voters: We set the price function for each voter i so that for $j \leq \lambda(i)$, shifting p by j positions costs j, and for $j > \lambda(i)$, shifting p by j positions costs $(2B+1)^j$ (where B is the total number of shift actions available; note that the exponential function $(2B+1)^j$ ensures that the price functions are convex and that we can easily identify situations where one shifts p by more than $\lambda(i)$ positions).[4]

Below we describe how to use this construction for the case of the Plurality rule and for the case of the Borda rule.

The Plurality Rule. We first translate the input instance into the non-combinatorial Plurality-Shift Bribery instance as described above. Then, we apply the known, exact, polynomial-time algorithm for the Plurality-Shift Bribery (Elkind et al., 2009) on this instance. Let s be the cost of the solution found for the non-combinatorial instance. If $s > B$, then it is impossible to ensure p's victory in the combinatorial instance (because the number of available shift actions is insufficient).

If $s \leq B$, then to obtain a solution F for the Plurality-Combinatorial Shift Bribery instance we do as follows. For each voter v that in the (non-combinatorial) bribed election ranks p first, we select shift actions in the combinatorial instance so that v ranks p first. Note that $|F| \leq s$ and that F is indeed a (combinatorial) solution.

For the sake of contradiction, assume that there is a successful set of shift actions F' with size smaller than $|F|/\Lambda$. However, it is easy to see that such a set of shift actions would correspond to a bribery of cost smaller than s for the non-combinatorial instance. Since s is the cost of the optimal solution for the non-combinatorial instance, this is a contradiction.

The Borda Rule. The case of Borda-Combinatorial Shift Bribery follows analogously, but instead of using the polynomial-time exact algorithm for the non-combinatorial instance, we use the 2-approximation algorithm for Borda-Shift Bribery (Elkind et al., 2009; Elkind & Faliszewski, 2010). Let s be the cost of the solution found. If $s > 2B$, then it is impossible to ensure p's victory.

Otherwise, to obtain the solution F for the combinatorial instance, for each vote v where the non-combinatorial solution shifts p by some t positions, we include t shift actions that affect this voter. We have that $|F| \leq s$, and F is a correct solution for the combinatorial instance.

If there existed a solution F' for the combinatorial instance that used less than $|F|/(2 \cdot \Lambda)$ shift functions, then there would be a solution for the non-combinatorial instance with cost smaller than $|F|/2 \leq s/2$. Since we used a 2-approximate algorithm for the non-combinatorial instance, this is impossible. □

We mention that it might be possible to improve the approximation ratio given in Theorem 14, at least for the Borda rule. The idea might be to cast the problem as a variant of

4. Strictly speaking, there is no need to ensure that the price functions are convex, but this is the variant of Shift Bribery that we generalize in this paper, so we stick to it for consistency.

the Set Multicover problem, which is a generalization of the Set Cover problem where each element has its own covering requirement. Then, one could use an approximation algorithm for the Set Multicover problem (for example, the one suggested in Rajagopalan & Vazirani, 1998) and plug it into the 2-approximation algorithm of Elkind and Faliszewski (2010).

We can achieve better approximation guarantees for the Borda rule, when we further restrict the allowed shift actions. To obtain these results we use the framework of Elkind and Faliszewski (2010). In essence, they have shown the following: If for a given variant of Shift Bribery, either for the Plurality rule or for the Borda rule, one can provide a function that computes how to obtain the highest number of points for the preferred candidate given some budget B, then there is a 2-approximation algorithm for this variant of Shift Bribery.[5] Note that such a get-most-points-for-p algorithm does not solve Shift Bribery. While it maximizes the score of p, it does not ensure that no candidate receives higher score. Indeed, an optimal solution might increase the score of p to a smaller extent, but at the expense of more dangerous opponents.

Theorem 15. *Borda-*Combinatorial Shift Bribery *is* 2*-approximable in polynomial time for* $(+1, +1)$*-shift actions.*

Proof. By the discussion preceding the theorem statement, it suffices to provide a function that given an instance of Combinatorial Shift Bribery with budget B finds a set of shift actions that obtain the highest possible number of points for the preferred candidate p without exceeding the budget.

The general idea of achieving this is to compute a maximum b-matching in an auxiliary multigraph (multigraphs allow multiple edges between the vertices). A b-matching of a multigraph G with a function $b : V(G) \to \mathbb{N}$ (called a *covering function*) is an edge-induced subgraph of G such that each vertex u has degree at most $b(u)$. It is known that a b-matching can be computed in polynomial time (Gabow, 1983).

We construct the auxiliary multigraph G as follows. For each voter v_i we create a vertex u_i. For each shift action with effect 1 on voter u_i and effect 1 on voter u_j, we create an edge $\{u_i, u_j\}$. Then, we define a covering function b such that $b(u_i)$ is the number of positions that p can be shifted forward in the preference order of voter v_i (that is, the position of p in the preference order of voter v_i).

If G has a b-matching of size at least B, then it corresponds to a set of shift actions that increase the score of p by $2B$, which is the highest gain possible. If G has a b-matching of size $k < B$, then we take the shift actions corresponding to the edges of this b-matching (these shift actions maximize the number of points that p can gain from shift actions that move p within two votes) and greedily select more shift actions that each pushes p forward in one vote, to use up the budget (at this point, every shift action can affect p in a single vote only). Thus our function computes the highest point gain possible for p, for a given budget. □

Next, we consider interval shift actions. That is, we fix some order of the voters and restrict each shift action to have effect only on voters which comprise intervals. (In fact, we

5. In fact, their result applies to all scoring rules, but in this paper we focus on the Plurality rule and on the Borda rule only.

could also allow "holes" inside these intervals.) Unfortunately, the algorithm requires XP time for the parameterization by the length of the longest interval.

Theorem 16. *For both the Plurality rule and the Borda rule,* Combinatorial Shift Bribery *can be* 2*-approximated in* XP*-time for interval shift actions, provided that we take* Λ*, the upper bound on the number of voters affected by each shift action, as the parameter.*

Proof. As per discussion preceding Theorem 15, it suffices to describe how to find a set of shift actions which maximize the number of points that the preferred candidate p gains under a given budget.

To this end, we use a dynamic programming algorithm. Consider an input for Combinatorial Shift Bribery with election $E = (C, V)$, preferred candidate p, and budget B to spend on increasing p's score. Let $m := |C|$ and $n := |V|$. We have $V = (v_1, \ldots, v_n)$. Our algorithm uses the following table for partial results. For numbers $x, y, s_0, \ldots, s_{\Lambda-1}$ the table entry:

$$T[x, y, s_0, s_1, \ldots, s_{\Lambda-1}]$$

denotes the maximum number of additional points that candidate p can gain from voters $v_1, \ldots, v_x$ under the condition that (1) exactly y shift actions are used, each of them affects at most the voters from the set $\{v_1, \ldots, v_x\}$, and (2) for each $i \in \{0, ..., \Lambda - 1\}$, candidate p is shifted to position s_i in the preference order of voter v_{x-i}. That is, we iterate over the voters and store the effect that the applied shift actions had on the last Λ voters. The size of the table is $n \cdot B \cdot m^{\Lambda+1}$.

Our algorithm is almost the same for both the Plurality rule and the Borda rule. The only difference is in computing the scores of the candidates. Let z, $0 \leq z \leq m - 1$, denote the position of p in the preference order of some voter (position 0 means that p is ranked first). Then, by score(z) we mean the score that p gains from this voter. For the Plurality rule we have score$(z) = 1$ for $z = 0$ and score$(z) = 0$ otherwise. For the Borda rule we have score$(z_i) = m - z_i - 1$. For a set of voters and a vector $z_1, \ldots, z_t$ (for $t \in [n]$ and each z_i in $\{0, \ldots, m - 1\}$) that denotes the positions of p in the preference orders of these voters, we write score$(z_1, \ldots, z_t)$ to mean the score that p gains from these voters. That is:

$$\text{score}(z_1, \ldots, z_t) = \sum_{i \in [t]} \text{score}(z_i).$$

Given this preparation, we are ready to describe our algorithm (jointly for the Plurality rule and for the Borda rule).

Initialization. We initialize the entries $T[\Lambda, y, s_0, s_1, \ldots, s_{\Lambda-1}]$ of the table as follows. We check whether there is a set of y shift actions that have effects only on voters from $(v_1, \ldots, v_\Lambda)$ and such that applying this set of y shift actions moves candidate p to positions $s_0, \ldots, s_{\Lambda-1}$ in the preference orders of the voters $v_1, \ldots, v_\Lambda$, respectively. If it exists, then we set $T[\Lambda, y, s_0, s_1, \ldots, s_{\Lambda-1}]$ to score$(s_0, s_1, \ldots, s_{\Lambda-1})$. Otherwise, we set $T[\Lambda, y, s_0, s_1, \ldots, s_{\Lambda-1}]$ to $-\infty$. (We explain how to check if such a set of shift actions exists at the end of the proof.)

Recursion Step. To compute the table entries $T[x, y, s_0, s_1, \ldots, s_{\Lambda-1}]$ for $x > \Lambda$, one has to compute subsets of i shift actions (for $i \in [y]$) whose last affected voter is v_x, that ensure—together with $y-i$ shift actions whose last affected voter is from the set $\{v_1, \ldots, v_{x-1}\}$—that for each j, $0 \leq j \leq \Lambda - 1$, p is shifted to position s_j in the preference order of v_{x-j}.

More specifically, in the update phase we compute for each x, $\Lambda < x \leq n$, each y, $0 \leq y \leq B$, and each vector $(s_0, \ldots, s_{\Lambda-1}) \in \{0, \ldots, m-1\}^\Lambda$ the table entry $T[x, y, s_0, s_1, \ldots, s_{\Lambda-1}]$ as follows. We say that a vector $(\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_{\Lambda-1}) \in \{0, \ldots, m\}^\Lambda$ is (x, i)-*realizable* for some i ($0 \leq i \leq y$), if there is a set of i shift actions whose last affected voter is v_x and such that for each j, $0 \leq j \leq \Lambda - 1$, it shifts candidate p by $\hat{s}_j$ positions in the preference order of voter v_{x-j}. We write $R(x, i)$ to denote the set of vectors from $\{0, \ldots, m-1\}^\Lambda$ that are (x, i)-realizable (we describe how to compute $R(x, i)$ later). Then, we compute $T[x, y, s_0, s_1, \ldots, s_{\Lambda-1}]$ as follows:

$$\begin{aligned} T[x, y, s_0, s_1, \ldots, s_{\Lambda-1}] = \max\{&T[x-1, y-i, s^*, s_0 - \hat{s}_1, \ldots, s_{\Lambda-1} - \hat{s}_{\Lambda-1}, s^*] \\ &+ \text{score}(s_0, s_1, \ldots, s_{\Lambda-1}) - \text{score}(s_1 - \hat{s}_1, \ldots, s_{\Lambda-1} - \hat{s}_{\Lambda-1}) \mid \\ &0 \leq i \leq y, 0 \leq s^* \leq m-1, (s_0, \hat{s}_1, \ldots, \hat{s}_{\Lambda-1}) \in R(x, i)\} \end{aligned}$$

Informally, for each "realizable total effect" of i shift actions whose last affected voter is v_x, the number of points that candidate p gains is the number of additional points that candidate p gains by shift actions for which the last affected voter is from $(v_1, \ldots, v_{x-1})$ plus the number of additional points that candidate p gains by shift actions for which the last affected voter is v_x (to avoid double counting, this is expressed as the difference in the middle line of the above formula).

We next show how to compute $R(x, i)$. We try every vector $(\hat{s}_0, \ldots, \hat{s}_{\Lambda-1}) \in \{0, \ldots, m-1\}^\Lambda$ and for each we check if it is (x, i)-realizable. Perhaps the easiest way to do this is to formulate this problem as an integer linear program (ILP) with a constant number of variables.

Let $(\hat{s}_0, \ldots, \hat{s}_{\Lambda-1})$ be a vector for which we want to check if it is (x, i)-realizable. For each subset $Q \subseteq \{0, \ldots, \Lambda - 1\}$, we say that a shift action is of type Q if it affects exactly the voters v_{x-i} with $i \in Q$. For each such subset Q, we introduce an integer variable x_Q, denoting the number of shift actions of type Q used in the (x, i)-realization of our vector. We solve the following ILP:

$$\sum_{Q \subseteq \{0, \ldots, \Lambda-1\}} x_Q = i \tag{1}$$

$$\sum_{Q \subseteq \{1, \ldots, \Lambda-1\}} x_{Q \cup \{0\}} = i \tag{2}$$

$$\sum_{j \in Q} x_Q = \hat{s}_j \qquad \forall j : 0 \leq j \leq \Lambda - 1 \tag{3}$$

(Note that the middle constraint ensures that the last affected voter is v_x.) Since the number of variables in this ILP is 2^Λ, it follows from the famous result of Lenstra (1983) that this ILP can be solved in XP time with respect to the parameter Λ (indeed, even in FPT time). Using the same ILP but without the middle constraint, we can check which vectors $(s_0, \ldots, s_{\Lambda-1})$ we can use in the initialization step.

Coming back to our dynamic program, it is clear that finding how to obtain the maximum score for p while respecting our budget can be found by taking the maximum over the table entries $T[n, B', s_0, s_1, \ldots, s_{\Lambda-1}]$, for all possible values of B', $0 \leq B' \leq B$, and $(s_0, s_1, \ldots, s_{\Lambda-1}) \in \{0, \ldots, m-1\}^{\Lambda}$. □

While in this section we showed that it is indeed possible to achieve some approximation algorithms for some special cases of the Combinatorial Shift Bribery problem, the settings for which our algorithms are efficient are quite restrictive. This means that in practice one might want to seek good heuristics and use our algorithms as a guidance for the initial search.

9. Conclusion

We have defined a combinatorial variant of the Shift Bribery problem (Elkind et al., 2009; Elkind & Faliszewski, 2010; Bredereck et al., 2014a) and we have studied its computational complexity. The motivation for our research was the desire to understand the computational difficulty imposed by correlated, large-scale effects of campaign actions. In this respect, this work is motivated by the combinatorial study of election control, as studied by Bulteau et al. (2015) and Chen et al. (2015). We have found that even for various very restricted special cases and numerous parameterizations, the Combinatorial Shift Bribery problem is highly intractable in the worst case. Nonetheless, we have found some initial positive results, mainly in the form of approximation algorithms. Interestingly, our approximation results quite strongly rely on the results for non-combinatorial Shift Bribery.

There is a number of research directions that are motivated by our work. For example, can Plurality-Combinatorial Shift Bribery or Borda-Combinatorial Shift Bribery be solved in polynomial-time for $(+1, +1)$ shift actions or interval actions under the assumption that the number of candidates is a constant?

More generally, our results suggest studying further restrictions of the problem. As an example, since parameterizing by the number of available shift actions immediately gives fixed-parameter tractability results, a natural question is whether other natural parameterizations exist which could also lead to positive results.

Naturally, one might consider other voting rules as well. Most interesting are Condorcet-consistent rules, such as the Copeland rule, since these rules tend to behave rather differently than scoring rules. We mention that some of our results do hold for other voting rules: specifically, Theorem 2, Theorem 5, Theorem 6, and Theorem 7 hold for most voting rules because these theorems hold for elections with only two candidates, and most voting rules behave the same for elections with only two candidates; Observation 1 and Theorem 12 are basically brute-force algorithms and the results hold for most voting rules as well; and the statements regarding the Borda rule in Theorem 14, Theorem 15, and Theorem 16 hold for all scoring rules, since the underlying 2-approximation algorithm of Elkind and Faliszewski (2010) works for all scoring rules.

Further, it might also be interesting to consider domain restrictions regarding voters' preferences (for example, single-crossing seems particularly natural in the context of interval shift actions, since it means that each shift action affects voters with somewhat similar preferences), as it is well-demonstrated that restricting the domain of the voters can lead

to tractability (see Theorem 10 in Bulteau et al., 2015, for an example in the combinatorial control setting). However, pursuing this direction would require a careful discussion of what shift actions can be applied. For example, should we allow a single-crossing election to cease being single-crossing after the bribery?

Acknowledgments

Robert Bredereck was supported by the DFG project PAWS (NI 369/10). Nimrod Talmon was supported by the DFG Research Training Group "Methods for Discrete Structures" (GRK 1408) and is currently at Weizmann Institute of Science. Piotr Faliszewski was supported by the DFG project PAWS (NI 369/10) and by AGH University grant 11.11.230.124 (statutory research).

A preliminary short version of this work has been presented at the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15) (Bredereck, Faliszewski, Niedermeier, & Talmon, 2015b).

Appendix A. Proof of Theorem 7

Theorem 7. *Unless* P = NP, Combinatorial Shift Bribery *is inapproximable (in polynomial time) for both the Plurality rule and the Borda rule, even for two candidates and* $(+1, -1)$*-shift actions.*

Proof. We give a many-one reduction from Set Cover. Let $(\mathcal{S}, X, h)$ be a Set Cover instance, with $\mathcal{S} = \{S_1, \ldots, S_m\}$ and $X = \{x_1, \ldots, x_n\}$ (we assume that every element belongs to at least one set). We construct an instance of Plurality-Combinatorial Shift Bribery. We set the budget $B := |X|$. The candidate set is $\{p, d\}$, where p is the preferred candidate. We have an *element voter* v_i for each element x_i, with preference order $d \succ p$. We have a *set voter* v_j^S for each set S_j, with preference order $p \succ d$. We also have $|X| + |\mathcal{S}| - 2h - 1$ *dummy voters*, each with preference order $d \succ p$. For each element x_i and each set S_j, if $x_i \in S_j$ then we construct a shift action f_j^i with effect of $+1$ on v_i and effect of -1 on v_j^S. This completes the construction. It is easy to see that it is computable in polynomial time.

Next, we show that there is a successful set of shift actions (note that the size of this set is not important, that is, we allow infinite budget) if and only if there is a set cover of size h.

For the "if" part, assume that there is a set cover $\mathcal{S}'$ of size at most h. We show how to build a successful set of shift actions. We start with $F' = \emptyset$ and for each element x_i, we choose an arbitrary set $S_j \in \mathcal{S}'$ which contains x_i and add the corresponding function f_j^i to F'. After applying F', observe that p becomes a winner: All $|X|$ element voters and $|\mathcal{S}| - h$ set voters prefer p but only $|X| + |\mathcal{S}| - 2h - 1$ dummy voters and h set voters prefer d.

For the "only if" part, assume that there is a successful set of shift actions $F' \subseteq F$. Let h' be the number such that after applying the shift actions from F', p is preferred by exactly $|S| - h'$ set voters (that is, shift actions in F' correspond to h' sets from $\mathcal{S}$). For p to be a winner, a majority of the voters (i.e., at least $|X| + |\mathcal{S}| - h$ voters) must prefer p. Thus, after applying F', at least $X - (h - h')$ element voters prefer p. This means that there is a collection of h' sets from $\mathcal{S}$ that jointly cover at least $|X| - (h - h')$ elements. Since every

element belongs to some set, we can extend this collection to a set cover by adding at most $h - h'$ sets (in the worst case, one set for each uncovered element). This proves that there is a set cover for $(\mathcal{S}, X, h)$ and completes the "only if" part.

Note that in the above argumentation we made no assumptions regarding the size of F'. Hence, finding any solution for our Plurality-Combinatorial Shift Bribery instance, including approximate solutions for any approximation factor, implies finding a set cover of size at most h. This means that unless $\mathsf{P} = \mathsf{NP}$, Plurality-Combinatorial Shift Bribery is inapproximable in polynomial time. □

Appendix B. Remaining Cases of the Proof of Theorem 8

Theorem 7. *Unless* $\mathsf{P} = \mathsf{NP}$, Combinatorial Shift Bribery *is inapproximable (in polynomial time) for both the Plurality rule and the Borda rule, even for two candidates and* $(+1, -1)$*-shift actions.*

The Borda Rule with $(+1, +1)$-Shift Actions. We can slightly modify the reduction used for the Plurality rule with $(+1, +1)$-shift actions. Specifically, we describe a parameterized reduction from the $\mathsf{W}[1]$-hard Clique problem, parameterized by the solution size, to Borda-Combinatorial Shift Bribery with $(+1, +1)$-shift actions, parameterized by (m, B).

Let (G, h) be an instance of Clique with $V(G) = \{u_1, \ldots, u_{n'}\}$ and $E(G) = \{e_1, \ldots, e_{m'}\}$. We create an instance of Borda-Combinatorial Shift Bribery as follows. The set of candidates is $\{p\} \cup D$, where $D = \{d_1, \ldots, d_{h-1}\}$. We create the following voters.

1. For each vertex $u_i \in V(G)$, we create a corresponding *vertex voter* v_i with preference order:
$$d_1 \succ \cdots \succ d_{h-1} \succ p.$$
2. We create $n' - 2h$ *dummy voters*, each with preference order:
$$p \succ d_2 \succ \cdots \succ d_{h-1} \succ d_1.$$
3. We create h *dummy voters*, each with preference order:
$$d_{h-1} \succ p \succ d_2 \succ \cdots \succ d_{h-2} \succ d_1.$$
4. We create $n' - h$ *dummy voters*, each with preference order:
$$p \succ d_1 \succ \cdots \succ d_{h-1}.$$
5. We create $n' - h$ *dummy voters*, each with preference order:
$$d_1 \succ p \succ d_2 \succ \cdots \succ d_{h-1}.$$

For each edge $\{u_i, u_j\} \in E(G)$, we create a shift action $f_{\{u_i,u_j\}}$ with effect 1 on the vertex voters v_i and v_j and effect 0 on all other voters. Finally, we set the budget to $B := \binom{h}{2}$. This completes the construction, which is computable in polynomial time.

The proof of correctness follows the same lines as the proof for the Plurality rule with $(+1, +1)$-shift actions, but instead of counting the number of approvals, we need to compute the Borda scores of the candidates. Indeed, this is the reason for our additional dummy voters.

In particular, the construction ensures that d_1 is the original winner of the election and that the difference between the Borda score of p and the Borda score of d_1 is exactly h^2. Furthermore, each shift action can increase the score of p by at most two. Hence, to make p a co-winner one must increase the score of p by $h(h-1)$ and decrease the score of d_1 by h. This is possible if and only if the shift actions correspond to edges of some clique of size h.

The Plurality Rule with $(+1, -1)$-Shift Actions. We still reduce from the W[1]-hard Clique problem, parameterized by the solution size, but the reduction is a bit more involved.

Let (G, h) be a Clique instance where the graph G has $n' := |V(G)|$ vertices and $m' := |E(G)|$ edges. We construct a Plurality-Combinatorial Shift Bribery instance as follows. Let the set of candidates be $\{p, d\} \cup D$, where $D := \{d_1, \ldots, d_{h-1}\}$, and create the following voters:

1. For each vertex v_i, create $\binom{h}{3}$ *vertex voters* $v_i^1, \ldots, v_i^{\binom{h}{3}}$ corresponding to v_i, each with preference order:
$$d_1 \succ \cdots \succ d_{h-1} \succ p.$$

2. For each edge $e_j = \{v_{i_1}, v_{i_2}\}$, create a corresponding *edge voter* w_j with preference order:
$$p \succ d_1 \succ \cdots \succ d_{h-1}.$$

3. Create $2\binom{h}{2} + (n' - 2h)\binom{h}{3} - m'$ *dummy voters*, each with preference order:
$$p \succ d_1 \succ \cdots \succ d_{h-1}.$$

For each edge $e_j = \{v_{i_1}, v_{i_2}\}$, construct $2\binom{h}{3}$ shift actions, denoted by
$$f^1_{e_j,v_{i_1}}, \ldots, f^{\binom{h}{3}}_{e_j,v_{i_1}} \text{ and } f^1_{e_j,v_{i_2}}, \ldots, f^{\binom{h}{3}}_{e_j,v_{i_2}},$$
where for each $z \in [\binom{h}{3}]$, (a) $f^z_{e_j,v_{i_1}}$ has effect of $+1$ on $v^z_{i_1}$ and effect of -1 on w_j, and (b) $f^z_{e_j,v_{i_2}}$ has effect of $+1$ on $v^z_{i_2}$ and effect of -1 on w_j. Finally, we set the budget $B := 2\binom{h}{2}\binom{h}{3}$. This completes the construction. It is easy to see that it is computable in polynomial time and that it is a parameterized reduction.

Observe that, initially, the edge voters and the dummy voters prefer p, while the vertex voters prefer d_1. Therefore, the initial score of p is $2\binom{h}{2} + (n' - 2h)\binom{h}{3}$, while the initial score of d_1 is $n'\binom{h}{3}$. We can assume, without loss of generality, that this means that d_1 is the winner of the election (instances not satisfying this assumption can be solved in constant time).

It remains to show that our constructed instance contains a successful set of shift actions F' of size at most h if and only if (G, h) contains a clique of size at most B. The general

idea is that if we choose the shift actions corresponding to the edges connecting the nodes of an h-size clique, then we will ensure that p becomes the preferred candidate for $h\binom{h}{3}$ additional vertex voters, while making d_1 the preferred candidate for only $\binom{h}{2}$ additional edge voters.

Formally, for the "if" part, let $H \subseteq V(G)$ be a set of h vertices forming a clique and let $E' \subseteq E(G)$ be the set of edges connecting vertices from H. We choose the following set of shift actions:

$$F' = \{f^z_{e_j,v_{i_1}}, f^z_{e_j,v_{i_2}} \mid e_j = \{v_{i_1}, v_{i_2}\} \in E', z \in [\binom{h}{3}]\}.$$

We show that F' is a successful set of shift actions. To this end, observe that for each vertex voter v_i^z with $v_i \in V'$ and $z \in [\binom{h}{3}]$, candidate p is shifted $h-1$ positions forward, therefore p becomes the preferred candidate for these voters. This means that $h\binom{h}{3}$ additional vertex voters prefer p (and, thus, do not prefer d_1 anymore). Furthermore, p is shifted backwards only for the voters in $\{w_j \mid e_j \in E'\}$, that is, d_1 becomes the most preferred candidate for $\binom{h}{2}$ edge voters and p remains the most preferred candidate for $m' - \binom{h}{2}$ edge voters. Thus, p and d tie as winners.

For the "only if" part, let F' be a successful set of shift actions. To make p a winner of the election, p must be shifted to the top position for at least $h\binom{h}{3} - \binom{h}{2}$ vertex voters (no other type of voters can be affected positively). By the pigeonhole principle, these vertex voters correspond to at least h different vertices (there are $\binom{h}{3}$ voters corresponding to each vertex). In effect, at least $\binom{h}{2}$ edge voters must be effected negatively so that d_1 becomes their most preferred candidate. Thus, to make p win the election p must be shifted to the top position for at least $h\binom{h}{3}$ vertex voters. This implies that $|F'| \geq (h-1)\cdot h\cdot\binom{h}{3} = 2\binom{h}{2}\binom{h}{3} = B$ and, hence, $|F'| = B$. It follows that p is shifted backwards making d_1 the most preferred candidate for exactly $\binom{h}{2}$ edge voters and that p must be shifted to the top position for exactly $h\binom{h}{3}$ vertex voters corresponding to exactly h different vertices. By construction, this implies that these h vertices form a clique, and we are done.

The Borda Rule with $(+1,-1)$-Shift Actions. For the Borda rule, the reduction is, once again, a bit more involved, but the main idea is the same as for the Plurality rule.

Let (G,h) be an instance of Clique where graph G has $n' := |V(G)|$ vertices and $m' := |E(G)|$ edges. We construct a Borda-Combinatorial Shift Bribery instance as follows. The set of candidates is $\{p,d\} \cup D$, where $D := \{d_1,\ldots,d_{h-1}\}$, and we create the following voters:

1. For each vertex v_i, create $\binom{h}{3}$ *vertex voters* $v_i^1,\ldots,v_i^{\binom{h}{3}}$ corresponding to v_i, each with preference order:
$$\overrightarrow{D} \succ p.$$

2. For each edge $e_j = \{v_{i_1}, v_{i_2}\}$, create a corresponding *edge voter* w_j with preference order:
$$d_1 \succ \cdots \succ d_{h-2} \succ p \succ d_{h-1}.$$

3. Let $L := \dfrac{\binom{h}{2} + (n'\binom{h}{3} + m')(h-1) - (\binom{h}{3}h^2) - m'}{h-1}$. Without loss of generality, we can assume that L is an integer (this requires simple modifications of the input clique

instance only). We create L *dummy voters*, each with preference order:

$$p \succ d_{h-1} \succ \cdots \succ d_1.$$

For each edge $e_j = \{v_{i_1}, v_{i_2}\}$, construct $2\binom{h}{3}$ shift actions, denoted by

$$f^1_{e_j,v_{i_1}}, \ldots, f^{\binom{h}{3}}_{e_j,v_{i_1}} \text{ and } f^1_{e_j,v_{i_2}}, \ldots, f^{\binom{h}{3}}_{e_j,v_{i_2}},$$

where for each $z \in [\binom{h}{3}]$, (a) $f^z_{e_j,v_{i_1}}$ has effect of $+1$ on $v^z_{i_1}$ and effect of -1 on w_j, and (b) $f^z_{e_j,v_{i_2}}$ has effect of $+1$ on $v^z_{i_2}$ and effect of -1 on w_j. Finally, we set the budget $B := 2\binom{h}{2}\binom{h}{3}$. This completes the construction. It is easy to see that it is computable in polynomial time. The proof of correctness follows the same lines as the proof of correctness for the Plurality rule and, thus, is omitted.

Appendix C. Proof of Theorem 9

Theorem 9. *Borda-*Combinatorial Shift Bribery *is* W[1]*-hard with respect to the number n of voters, even for $(+1, -1)$-shift actions and no budget constraints.*

Proof. We reduce from the following W[1]-hard problem (Mathieson & Szeider, 2012, Lemma 3.2).

> Strongly Regular Multicolored Clique
> **Input:** Two integers, d and h, and an undirected graph $G = (V, E)$, where each vertex has one of h colors in $[h]$, and where each vertex is adjacent to exactly d vertices of each color different from its own.
> **Question:** Does there exist a clique of size h containing one vertex from each color class?

Given an instance of Strongly Regular Multicolored Clique, we construct an instance of Combinatorial Shift Bribery, for the Borda rule. The general idea of the reduction is as follows. The set of "important" candidates consists of our preferred candidate p and the candidates that correspond to the edges. For technical reasons, for each edge $e = \{v, v'\}$, we introduce two candidates, e^1 and e^2; one of them is associated with "touching" vertex v and the other is associated with "touching" vertex v'. (In fact, we will introduce more edge candidates and some vertex candidates, but we will use them only to ensure correct structure of the election and appropriate bribery behavior.) We build two groups of voters, the vertex-selecting voters and the edge-electing voters. The first group implements picking vertices for the clique (one vertex from each color), and the second group implements picking edges (one edge for each pair of colors). We ensure that if a given set of shift actions has any chance of being successful, then it must hold that h vertices and $\binom{h}{2}$ edges are picked. Importantly, this holds even in the unbribed election.

We make sure that p wins the election if and only if the picked voters and edges correspond to a clique (with vertices of each color). To this end, we define the voters so that there are two numbers, α and β, such that:

1. There are h vertices picked by the vertex-selecting voters, each of a different color. The vertex-selecting voters give α points to each edge candidate that is associated with touching one of the selected vertices, and $\alpha + 1$ points to all other edge candidates. This means that by picking a vertex we decrease the score of the edge candidates for the edges that touch this vertex.

2. There are $\binom{h}{2}$ edges picked by the edge-selecting voters, one edge for each pair of colors. The edge-selecting voters give $\beta + 1$ points to each edge candidate that corresponds to a picked edge, and β points to all the remaining edge candidates. This means that, by picking an edge, we increase the score of the candidates corresponding to it.

3. Candidate p gets $\alpha + \beta + 1$ points, irrespective what shift actions we apply.

Note that in the unbribed election every candidate gets at most $\alpha + \beta + 2$ points and p always gets $\alpha + \beta + 1$ points. Thus the challenge is to ensure that every candidate gets $\alpha + \beta + 1$ points. By the above description, this is possible only if we pick the vertices and the edges that correspond to a size h clique (of vertices with different colors). Indeed, if we selected an edge e that did not touch two selected vertices, then e^1 and e^2 would receive $\beta + 1$ points from edge-selecting candidates and at least one of them would receive $\alpha + 1$ points from vertex-selecting voters. In effect, p would not be a winner.

Without loss of generality, we assume that the edges and vertices selected in the unbribed election do not form a clique (otherwise there would be a trivial solution for the input problem and we could output a fixed "yes"-instances of Borda-Combinatorial Shift Bribery).

Construction. We now formally describe the reduction, then we give an example of applying it to a simple instance, and finally we show correctness of the reduction. We illustrate some aspects of the correctness proof using our example.

Candidates. Our set of candidates is somewhat involved. Our important candidates are the preferred candidate p and the sets of edge candidates, E_1 and E_2, defined below.

Let $E(G) = \{e_1, \ldots, e_\mu\}$ be the set of edges of graph G. We create two edge-candidate sets: $E^1 = \{e_1^1, \ldots, e_\mu^1\}$ and $E^2 = \{e_1^2, \ldots, e_\mu^2\}$. For each $i \in [h]$, let n_i be the number of vertices in G with color i and let $V^i = \{v_1^i, \ldots, v_{n_i}^i\}$ be the set of these vertices. For each color i and each vertex $v_j^i \in V^i$, we define the neighborhood of v_j^i as follows:

$$N(v_j^i) := \{e_\ell^1 \mid e_\ell = \{v_j^i, v_{j'}^{i'}\} \in E \wedge i < i'\}$$
$$\cup \{e_\ell^2 \mid e_\ell = \{v_j^i, v_{j'}^{i'}\} \in E \wedge i > i'\}.$$

(This, perhaps a bit strange way of using color numbers to pick edge candidates either from E^1 or E^2, is implementing the fact that for each edge $e \in E(G)$ we have two candidates, e^1 and e^2, associated with touching different endpoints of e.)

For technical reasons we need further candidates as follows. To adjust the scores of all other candidates, we introduce a single dummy candidate d. We create two further candidate sets $E^0 = \{e_1^0, \ldots, e_\mu^0\}$ and $E^3 = \{e_1^3, \ldots, e_\mu^3\}$ which will act as "guards" for the edge-selecting voters. For each V^i we create two candidate sets $U^i := \{u_j^i \mid v_j^i \in V^i\}$ and

$U'^i = \{u'^i_j \mid v^i_j \in V^i\}$ with $U := \bigcup_{1\le i\le h} U^i$ and $U' := \bigcup_{1\le i\le h} U'^i$ which will act as "guards" for the vertex-selecting voters.

Our final set of candidates is $C := U \cup U' \cup E^0 \cup E^1 \cup E^2 \cup E^3 \cup \{p, d\}$.

Vertex-Selecting Voters. We now describe the vertex-selecting voters. For each color i and each vertex v^i_j, we define the following parts of preference orders (for $j = 1$, we assume that u^i_{j-1} and u'^i_{j-1} are $u^i_{n_i}$ and $u'^i_{n_i}$ respectively):

$$A(v^i_j) : u^i_j \succ \overrightarrow{N(v^i_j)} \succ u'^i_j,$$
$$B(v^i_j) : u^i_{j-1} \succ \overrightarrow{N(v^i_j)} \succ u'^i_{j-1}.$$

For each color i we create three pairs of voters. The voters in the first pair, w_i and w'_i, have the following preference orders:

$$w_i : p \succ A(v^i_1) \succ A(v^i_2) \succ A(v^i_3) \succ \cdots \succ A(v^i_{n_i}) \succ \overrightarrow{R^i},$$
$$w'_i : \overrightarrow{R^i} \succ B(v^i_1) \succ B(v^i_{n_i}) \succ B(v^i_{n_i-1}) \succ \cdots \succ B(v^i_2) \succ p,$$

where R^i is the set of the remaining candidates, that is, $R^i := C \setminus (\{p\} \cup U^i \cup U'^i \cup N(v^i_1) \cup \cdots \cup N(v^i_{n_i}))$. The voters in the second pair, q_i and q'_i, have preference orders that are the reverse of w_i and the reverse of w'_i, respectively. Finally, the voters in the last pair, $\bar{q}_i$ and $\bar{q}'_i$, have preference orders:

$$\bar{q}_i : \overrightarrow{C \setminus (\{d\} \cup N(v^i_1))} \succ d \succ \overrightarrow{N(v^i_1)},$$
$$\bar{q}'_i : \overleftarrow{N(v^i_1)} \succ \overleftarrow{C \setminus (\{d\} \cup N(v^i_1))} \succ d.$$

In effect, the first two pairs of voters jointly give $2(|C|-1)$ points to each of the candidates. The last pair gives $|C|-1$ points to the candidates in $N(v^i_1)$ and $|C|$ points to all other candidates (except d, who receives less than $|C|-1$ points).

Let $\alpha := h(2(|C|-1)+|C|)-1$. Altogether, the vertex-selecting voters give the following scores to the candidates: The candidates in $N(v^1_1) \cup N(v^2_1) \cup \cdots \cup N(v^h_1)$ receive α points and all other candidates, except d, receive $\alpha+1$ points (d receives less than α points). Thus, in the unbribed election $v^1_1, \ldots, v^h_1$ are the selected vertices.

For each color i, we introduce $(n_i - 1) \cdot ((h-1) \cdot d + 2)$ shift actions with effect -1 on voter w_i and effect $+1$ on voter w'_i. To understand where the number of these shift actions comes from, we note that: (1) For each vertex v^i_j, we have $|N(v^i_j)| = (h-1) \cdot d$ (each vertex is connected with d vertices of each color different than its own), (2) in $A(v^i_j)$ and in $B(v^i_j)$ the candidates from $N(v^i_j)$ are surrounded by two vertex candidates, and (3) if t is an integer, $1 \le t \le n_i - 1$, then applying $t((h-1) \cdot d + 2)$ of these shift actions has the effect that the candidates in $N(v^i_1)$ gain one point (i.e., v^i_1 ceases to be selected), the candidates in $N(v^i_{t+1})$ lose one point (i.e., v^i_{t+1} becomes selected), and no other candidate changes his score (later we will argue that applying other numbers of such shift actions than multiples of $((h-1) \cdot d + 2)$ cannot ensure p's victory).

Edge-Selecting Voters. For the edge-selecting voters, we need the following additional notation. Let $E_{x,y}$ denote the set of candidates representing edges between vertices of color x and color y, that is,

$$E_{x,y} := \{e_\ell^{q\in\{0,1,2,3\}} \mid e_\ell = \{v_j^x, v_{j'}^y\} \in E\}.$$

We write $n_{x,y}$ to denote the number of edges between vertices of color x and color y. By $\mathrm{id}_z^{x,y}$ we refer to the index of the z-th edge between vertices of color x and y. For example, if e_3, e_7 and e_{57} are the only three edges between vertices of colors 1 and 2, then $n_{1,2} = 3$, $\mathrm{id}_1^{1,2} = 3$, $\mathrm{id}_2^{1,2} = 7$, $\mathrm{id}_3^{1,2} = 57$.

For each pair $\{x, y\}$ of distinct colors and each edge $e_{\mathrm{id}_j^{x,y}}$, we introduce the following parts of preference orders (for $j = n_{x,y}$, we assume that $\mathrm{id}_{j+1}^{x,y} = \mathrm{id}_1^{x,y}$):

$$R(e_{\mathrm{id}_j^{x,y}}) : e^0_{\mathrm{id}_j^{x,y}} \succ e^1_{\mathrm{id}_j^{x,y}} \succ e^2_{\mathrm{id}_j^{x,y}} \succ e^3_{\mathrm{id}_j^{x,y}},$$
$$S(e_{\mathrm{id}_j^{x,y}}) : e^0_{\mathrm{id}_{j+1}^{x,y}} \succ e^1_{\mathrm{id}_j^{x,y}} \succ e^2_{\mathrm{id}_j^{x,y}} \succ e^3_{\mathrm{id}_{j+1}^{x,y}}.$$

For each pair $\{x, y\}$ of distinct colors we introduce three pairs of voters. The voters in the first pair, $w_{x,y}$ and $w'_{x,y}$, have the following preference orders:

$$w_{x,y} : R(e_{\mathrm{id}_1^{x,y}}) \succ p \succ R(e_{\mathrm{id}_2^{x,y}}) \succ R(e_{\mathrm{id}_3^{x,y}}) \succ \cdots \succ R(e_{\mathrm{id}_{n_{x,y}}^{x,y}}) \succ \overrightarrow{R^{x,y}},$$
$$w'_{x,y} : \overrightarrow{R^{x,y}} \succ S(e_{\mathrm{id}_{n_{x,y}}^{x,y}}) \succ S(e_{\mathrm{id}_{n_{x,y}-1}^{x,y}}) \succ \cdots \succ S(e_{\mathrm{id}_2^{x,y}}) \succ S(e_{\mathrm{id}_1^{x,y}}) \succ p,$$

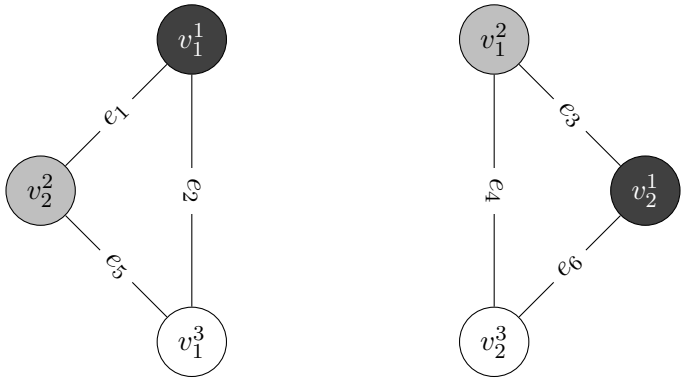
where $R^{x,y}$ is the set of the remaining candidates, that is, $R^{x,y} := C \setminus (\{p\} \cup E_{x,y})$. The voters in the second pair, $q_{x,y}$ and $q'_{x,y}$, have preference orders that are the reverse of $w_{x,y}$ and the reverse of $w'_{x,y}$, respectively. Finally, the voters in the last pair, $\bar{q}_{x,y}$ and $\bar{q}'_{x,y}$, have the following preference orders:

$$\bar{q}_{x,y} : e^1_{\mathrm{id}_1^{x,y}} \succ e^2_{\mathrm{id}_1^{x,y}} \succ d \succ \overrightarrow{C \setminus (\{d, e^1_{\mathrm{id}_1^{x,y}}, e^2_{\mathrm{id}_1^{x,y}}\})},$$
$$\bar{q}'_{x,y} : \overleftarrow{C \setminus (\{d, e^1_{\mathrm{id}_1^{x,y}}, e^2_{\mathrm{id}_1^{x,y}}\})} \succ e^2_{\mathrm{id}_1^{x,y}} \succ e^1_{\mathrm{id}_1^{x,y}} \succ d.$$

The first two pairs of voters jointly give $2(|C| - 1)$ points to each of the candidates. The last pair gives $|C|$ points to both $e^1_{\mathrm{id}_1^{x,y}}$ and $e^2_{\mathrm{id}_1^{x,y}}$, and $|C| - 1$ points to all other candidates (except d, who receives less than $|C| - 1$ points).

Let $\beta := 3\binom{h}{2}(|C| - 1)$. Altogether, for each pair of distinct colors $\{x, y\}$, the edge-selecting voters give $\beta + 1$ points to candidates $e^1_{\mathrm{id}_1^{x,y}}$ and $e^2_{\mathrm{id}_1^{x,y}}$. All other candidates receive β points (except for d, who receives less than β points). Thus in the unbribed election the selected edges are exactly "the first edges between each pair of colors" (that is, edges of the form $e_{\mathrm{id}_1^{x,y}}$, for each pair of distinct colors $\{x, y\}$).

For each pair $\{x, y\}$ of distinct colors, we create $4(n_{x,y} - 1)$ shift actions with effect -1 on voter $w_{x,y}$ and effect $+1$ on voter $w'_{x,y}$. The intuition behind these shift actions is similar as in the case of vertex-selecting voters. We make the following observations: (1) For each edge $e_{\mathrm{id}_\ell^{x,y}}$ there are four candidates listed in $R(e_{\mathrm{id}_\ell^{x,y}})$ and four candidates listed in $S(e_{\mathrm{id}_\ell^{x,y}})$, and (2) if t is an integer, $1 \le t \le n_{x,y} - 1$, and we apply $4t$ such shift actions, then candidates



$V^1 = \{v_1^1, v_2^1\}$, $V^2 = \{v_1^2, v_2^2\}$, $V^3 = \{v_1^3, v_2^3\}$, $h = 3$, $d = 1$

Figure 2: A 3-colored graph with six vertices where each vertex is adjacent to one vertex from each of the color classes V^1, V^2 and V^3, other than its own.

$e^1_{\mathrm{id}_1^{x,y}}$ and $e^2_{\mathrm{id}_1^{x,y}}$ lose one point (edge $e_{\mathrm{id}_1^{x,y}}$ ceases to be selected), candidates $e^1_{\mathrm{id}_{t+1}^{x,y}}$ and $e^2_{\mathrm{id}_{t+1}^{x,y}}$ gain one point (edge $e_{\mathrm{id}_{t+1}^{x,y}}$ becomes selected), and the scores of all other candidates remain unchanged (we will later argue that if we apply a number of shift actions that is not a multiple of 4, then p certainly is not a winner of the resulting election).

To conclude the construction, we set the budget $B := \infty$ (that is, we can use as many shift actions as we like). It is easy to verify that the reduction is computable in polynomial time and that we introduce a number of voters that is a function of h only (thus, it is a parameterized reduction). Before proving the correctness of this construction, we consider the following example (we will refer to it during the correctness proof as well).

Example 5. Consider the Strongly Regular Multicolored Clique instance (d, h, G) with $d = 1$, $h = 3$, and graph G from Figure 2. Our construction produces the following set of candidates:

$$C := U \cup U' \cup E^0 \cup E^1 \cup E^2 \cup E^3 \cup \{p, d\},$$

with

$$U = \{u_1^1, u_2^1, u_1^2, u_2^2, u_1^3, u_2^3\},\ U' = \{u_1'^1, u_2'^1, u_1'^2, u_2'^2, u_1'^3, u_2'^3\}$$

and

$$E^i = \{e_1^i, e_2^i, \ldots, e_6^i\}, 0 \leq i \leq 3.$$

Furthermore, we have:

$$\begin{aligned} N(v_1^1) &:= \{e_1^1, e_2^1\}, & N(v_2^1) &:= \{e_3^1, e_6^1\}, \\ N(v_1^2) &:= \{e_3^2, e_4^1\}, & N(v_2^2) &:= \{e_1^2, e_5^1\}, \\ N(v_1^3) &:= \{e_2^2, e_5^2\}, & N(v_2^3) &:= \{e_4^2, e_6^2\}. \end{aligned}$$

For the vertex-selecting group of voters, we create the following voters. For each color i, we create two voters w_i and w'_i:

$$\begin{aligned}
w_1 &: p \succ u_1^1 \succ e_1^1 \succ e_2^1 \succ u_1'^1 \succ u_2^1 \succ e_3^1 \succ e_6^1 \succ u_2'^1 \succ \overrightarrow{R^1},\\
w'_1 &: \overrightarrow{R^1} \succ u_2^1 \succ e_1^1 \succ e_2^1 \succ u_2'^1 \succ u_1^1 \succ e_3^1 \succ e_6^1 \succ u_1'^1 \succ p,\\
w_2 &: p \succ u_1^2 \succ e_3^2 \succ e_4^1 \succ u_1'^2 \succ u_2^2 \succ e_1^2 \succ e_5^1 \succ u_2'^2 \succ \overrightarrow{R^2},\\
w'_2 &: \overrightarrow{R^2} \succ u_2^2 \succ e_3^2 \succ e_4^1 \succ u_2'^2 \succ u_1^2 \succ e_1^2 \succ e_5^1 \succ u_1'^2 \succ p,\\
w_3 &: p \succ u_1^3 \succ e_2^2 \succ e_5^2 \succ u_1'^3 \succ u_2^3 \succ e_4^2 \succ e_6^2 \succ u_2'^3 \succ \overrightarrow{R^3},\\
w'_3 &: \overrightarrow{R^3} \succ u_2^3 \succ e_2^2 \succ e_5^2 \succ u_2'^3 \succ u_1^3 \succ e_4^2 \succ e_6^2 \succ u_1'^3 \succ p,
\end{aligned}$$

with $R^i := C \setminus (\{p\} \cup U^i \cup U'^i \cup N(v_1^i) \cup \cdots \cup N(v_{n_i}^i))$, $1 \le i \le 3$. For each of these voters we add a voter with reversed preferences. (This means that, so far, all candidates obtain the same total score.) We finish this group of voters by creating for each color i two voters, $\bar{q}_i$ and $\bar{q}'_i$, with preference orders:

$$\begin{aligned}
\bar{q}_i &: \overrightarrow{C \setminus (\{d\} \cup N(v_1^i))} \succ d \succ \overrightarrow{N(v_1^i)},\\
\bar{q}'_i &: \overleftarrow{N(v_1^i)} \succ \overleftarrow{C \setminus (\{d\} \cup N(v_1^i))} \succ d.
\end{aligned}$$

This ensures that for each color i, all the candidates in $N(v_1^i)$ get α points, and all other candidates get $\alpha + 1$ points (except d who gets few points). We create 4 shift actions with effect -1 on voter w_i and effect $+1$ on voter w'_i.

For the edge-selecting second group of voters, recall that $E_{x,y}$ denotes the set of candidates representing edges between vertices of color x and color y. Specifically, we have:

$$\begin{aligned}
E_{1,2} :=& \{e_1^0, e_1^1, e_1^2, e_1^3\, e_3^0, e_3^1, e_3^2, e_3^3\},\\
E_{1,3} :=& \{e_2^0, e_2^1, e_2^2, e_2^3\, e_6^0, e_6^1, e_6^2, e_6^3\}, \text{ and}\\
E_{2,3} :=& \{e_4^0, e_4^1, e_4^2, e_4^3\, e_5^0, e_5^1, e_5^2, e_5^3\}.
\end{aligned}$$

For each pair $\{x, y\}$ of distinct colors we create two voters, $w_{x,y}$ and $w'_{x,y}$, as follows:

$$\begin{aligned}
w_{1,2} &: e_1^0 \succ e_1^1 \succ e_1^2 \succ e_1^3 \succ p \succ e_3^0 \succ e_3^1 \succ e_3^2 \succ e_3^3 \succ \overrightarrow{R^{1,2}},\\
w'_{1,2} &: \overrightarrow{R^{1,2}} \succ e_1^0 \succ e_3^1 \succ e_3^2 \succ e_1^3 \succ e_3^0 \succ e_1^1 \succ e_1^2 \succ e_3^3 \succ p,\\
w_{1,3} &: e_2^0 \succ e_2^1 \succ e_2^2 \succ e_2^3 \succ p \succ e_6^0 \succ e_6^1 \succ e_6^2 \succ e_6^3 \succ \overrightarrow{R^{1,3}},\\
w'_{1,3} &: \overrightarrow{R^{1,3}} \succ e_2^0 \succ e_6^1 \succ e_6^2 \succ e_2^3 \succ e_6^0 \succ e_2^1 \succ e_2^2 \succ e_6^3 \succ p,\\
w_{2,3} &: e_4^0 \succ e_4^1 \succ e_4^2 \succ e_4^3 \succ p \succ e_5^0 \succ e_5^1 \succ e_5^2 \succ e_5^3 \succ \overrightarrow{R^{2,3}},\\
w'_{2,3} &: \overrightarrow{R^{2,3}} \succ e_4^0 \succ e_5^1 \succ e_5^2 \succ e_4^3 \succ e_5^0 \succ e_4^1 \succ e_4^2 \succ e_5^3 \succ p,
\end{aligned}$$

where $R^{x,y} := C \setminus (\{p\} \cup E[x, y])$. For each of these voters we add a voter with reversed preferences. Further, for each pair $\{x, y\}$ of distinct colors, we add two voters $\bar{q}_{x,y}$ and $\bar{q}'_{x,y}$

as follows:

$$\bar{q}_{x,y} : e^1_{\mathrm{id}_1^{x,y}} \succ e^2_{\mathrm{id}_1^{x,y}} \succ d \succ \overrightarrow{C \setminus (\{d, e^1_{\mathrm{id}_1^{x,y}}, e^2_{\mathrm{id}_1^{x,y}}\})},$$
$$\bar{q}'_{x,y} : \overleftarrow{C \setminus (\{d, e^1_{\mathrm{id}_1^{x,y}}, e^2_{\mathrm{id}_1^{x,y}}\})} \succ e^2_{\mathrm{id}_1^{x,y}} \succ e^1_{\mathrm{id}_1^{x,y}} \succ d.$$

Altogether, for each pair $\{x, y\}$ of distinct colors, candidates $e^1_{\mathrm{id}_1^{x,y}}$ and $e^2_{\mathrm{id}_1^{x,y}}$ get $\beta+1$ points and all other candidates get β points (except d, which gets less points). For each pair $\{x, y\}$ of distinct colors, we create 4 shift actions with effect -1 on voter $w_{x,y}$ and effect $+1$ on voter $w'_{x,y}$. △

Properties of the Construction. We now discuss several properties of our construction. These properties will play a significant rule in showing the correctness of the reduction. To illustrate our arguments, we come back to our example from time to time. We begin by looking at the scores of the candidates.

Lemma 1. *The following claims hold:*

1. *In the unbribed election, every candidate receives at most $\alpha + \beta + 2$ points and every candidate from $\{p\} \cup U \cup U' \cup E^0 \cup E^3$ receives exactly $\alpha + \beta + 1$.*
2. *In every bribed election, the score of p is exactly $\alpha + \beta + 1$.*
3. *After applying a successful set of shift actions, the score of p is $\alpha + \beta + 1$ and the scores of all other candidates are at most $\alpha + \beta + 1$.*

Proof. It is easy to see that the first claim holds based on the discussion that we give throughout the construction. The second claim holds because (a) applying every shift action decreases by one the score of p in one vote and increases it by one in another vote (there are sufficiently few shift actions in the whole instance such that applying each shift action always moves p within the two votes on which the shift action acts). The last claim follows directly from the second one. (Lemma) □

Let us now consider the process of selecting vertices. In the description of vertex-selecting voters we said that, initially, for each color i vertex v^i_1 is selected, and if for some integer t, $1 \leq i \leq n_i - 1$, we apply $t((h-1) \cdot d + 2)$ shift actions that affect voters w_i and w'_i, then v^i_1 ceases to be selected and v^i_{t+1} becomes selected. We now argue that if we apply a number of these shift actions that is not divisible by $((h-1) \cdot d + 2)$, then p is not a winner of the resulting election.

To see that this is the case, recall that in the preference orders of voter w_i and w'_i there are exactly $(h-1) \cdot d$ candidates from $E^1 \cup E^2$ between each pair of candidates $\{u^i_j, u'^i_j\}$. Furthermore, if p passes some candidate u^i_j in the preference order of voter w_i (increasing u^i_j's score by one), then it must also pass candidate u^i_j in the preference order of voter w'_i (decreasing u^i_j's score by one). Otherwise, u^i_j would end up with score $\alpha + \beta + 2$ and, by 1, p would not be a winner (there are no possibilities to influence the score of u^i_j other than shifting p in the preference lists of w_i and w'_i). Hence, p also passes candidate u'^i_j and all candidates between u^i_j and u'^i_j in the preference lists of w_i and w'_i. This, however, means that if p is to be a winner of the election, then the number of applied shift actions

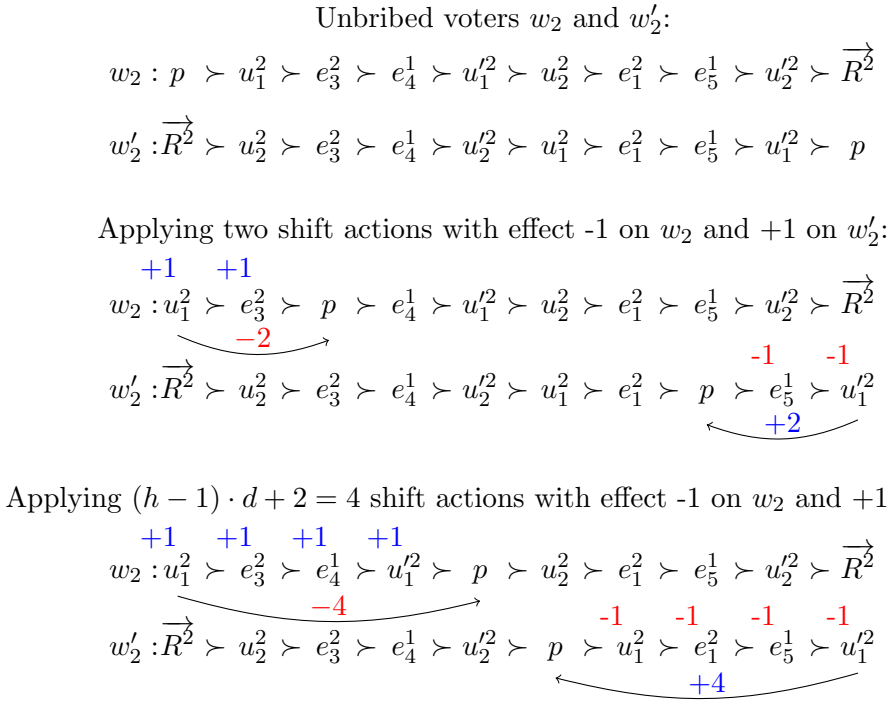


Figure 3: Illustration of bribery actions affecting the first voter group of our running example (Example 5). Note that, in the unbribed election, every candidate from $U \cup U'$ obtains $\alpha + \beta + 1$ points in total. For each color i there is only one type of shift actions which affects voter w_i and w_i': those shift actions with effect -1 on voter w_i and effect $+1$ on voter w_i'. No other shift action can affect some voter from the first group. Applying a multiple of $((h-1)\cdot d+2)$ shift actions with effect -1 on voter w_i and effect $+1$ on voter w_i' ensures that the candidates from $U^i \cup U'^i$ receive at most $\alpha+\beta+1$ points in total, whereas applying any other number of these shift actions implies that some candidate from U^i receives $\alpha+\beta+2$ points and, hence, p cannot win. We illustrate this with color 2 in our running example.

with effects on voters w_i and w_i' is a multiple of $((h-1)\cdot d+2)$ (p passes candidate u_j^i, candidate $u_j'^i$, and $h \cdot d$ candidates in between). Figure 3 provides an illustration of the above reasoning.

We next discuss selecting edges. As for the case of vertex-selecting voters, in the description of our construction we have argued that (a) initially for each pair $\{x, y\}$ of distinct colors, edge $e_{\mathrm{id}_1^{x,y}}$ is selected and that (b) after applying $4t$, $1 \leq t \leq n_{x,y} - 1$, shift actions that affect voters $w_{x,y}$ and $w'_{x,y}$, $e_{\mathrm{id}_1^{x,y}}$ ceases to be selected and $e_{\mathrm{id}_{t+1}^{x,y}}$ becomes selected. We now argue that if we used a number of such shift actions that is not a multiple of four, then p certainly would not be a winner of the election.

To see that this is the case, note that we designed the preference orders of $w_{x,y}$ and $w'_{x,y}$ so that the candidates $e^0_{\mathrm{id}_j^{x,y}}$ and $e^3_{\mathrm{id}_j^{x,y}}$, for $j \in \{2, \ldots, n_{x,y}\}$, follow p in vote $w_{x,y}$ in the same order in which they precede p in $w'_{x,y}$. In effect, if we apply a shift action that affects

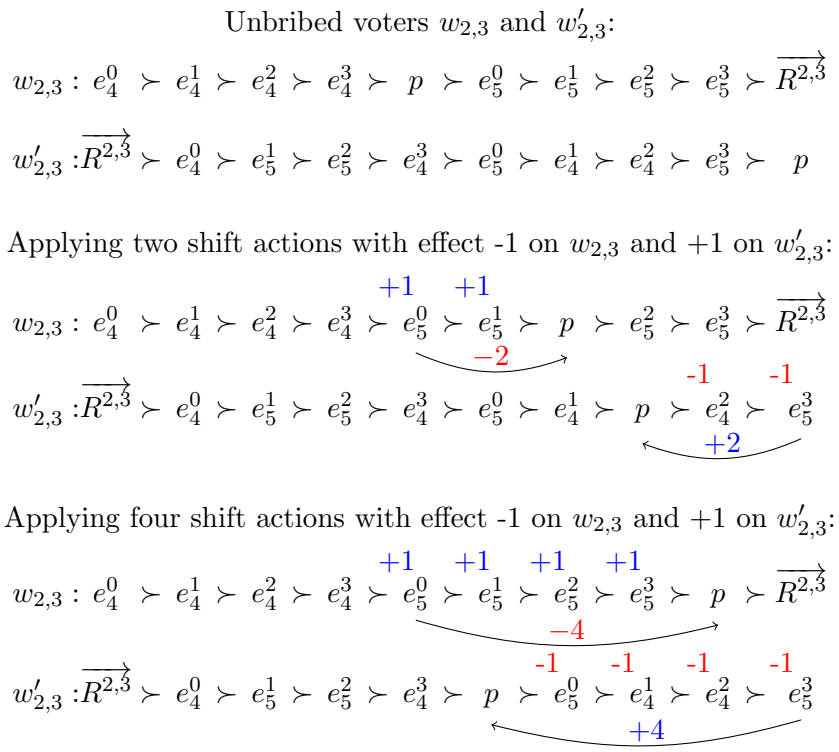


Figure 4: Illustration of bribery actions affecting the second voter group of our running example. Note that in the unbribed election, every candidate from $E^0 \cup E^4$ obtains $\alpha + \beta + 1$ points in total. For each pair of colors x and y there is only one type of shift actions which affects voter $w_{x,y}$ and $w'_{x,y}$: those shift actions with effect -1 on voter $w_{x,y}$ and effect $+1$ on voter $w'_{x,y}$. No other shift action can affect some voter from the second group. Applying a multiple of 4 shift actions with effect -1 on voter $w_{x,y}$ and effect $+1$ on voter $w'_{x,y}$ ensures that the candidates from $E^0 \cup E^4$ receive at most $\alpha + \beta + 1$ points from these voters, whereas applying any other number of these shift actions implies that some candidate from E^0 receives $\alpha + \beta + 2$ points and, hence, p cannot win. We illustrate this with color pair 2 and 3 in our running example.

voters $w_{x,y}$ and $w'_{x,y}$ a number of times that is not a multiple of four, then one of these candidates obtains $\alpha + \beta + 2$ points. Since there is no other way to affect the score of these candidates, by Lemma 1, in this case p cannot be a winner. We illustrate this effect in Figure 4.

Solution for Example 5. Before we complete the correctness proof, let us illustrate the solution for our example.

The unbribed election selects vertex v_1^1, v_1^2, and v_1^3 and the edges e_1, e_2 and e_4. Hence, for example, candidate e_4^2 receives $\alpha+\beta+2$ points and p (who receives only $\alpha+\beta+1$ points) is not a winner.

By applying four shift actions with effect -1 on w_2 and effect $+1$ on w'_2, we select v_2^2 instead of v_1^2 to be the vertex of color 2 in our clique (as depicted in the bottom of Figure 3).

By applying four shift actions with effect -1 on $w_{2,3}$ and effect $+1$ on $w'_{2,3}$, we select e_5 instead of e_4 to be the edge between color 2 and color 3 in our clique (as depicted in the bottom of Figure 4). Now, each candidate from $\{e_1^1, e_2^1, e_1^2, e_2^2, e_5^1, e_5^2\}$ receives $\beta + 1$ points from the edge-selecting voters, but only α points from the vertex-selecting voters. Every other candidate receives at most $\alpha + 1$ points from the vertex-selecting voters and at most β points from the edge-selecting voters. Hence, p (with $\alpha + \beta + 1$ points) is a winner. This solution corresponds to the left 3-colored triangle in Figure 2.

Correctness. It remains to show that there is a successful set of shift actions for the constructed Borda-Combinatorial Shift Bribery instance if and only if there is an h-colored clique in graph G.

For the "if" part, assume that there is an h-colored clique $H \subseteq V(G)$. Without loss of generality, let $H = \{v_{z_1}^1, \ldots, v_{z_h}^h\}$ and let $E_H := \{\{v, v'\} \mid v, v' \in H\}$. Furthermore, let $z_{x,y}$ denote the index of the edge in $E_{x,y}$ representing the edge from E_H between the vertex of color x and the vertex of color y. That is, $z_{x,y} = j$ if and only if $e_{\mathrm{id}_j^{x,y}} \in E_H$. Then it is easy to verify that the following set of shift actions is successful:

1. For each color $i \in [h]$, include $(z_i - 1)((h-1) \cdot d + 2)$ shift actions with effects on voters w_i and w'_i.

2. For each pair $\{x, y\}$ of distinct colors, include $4(z_{x,y} - 1)$ shift actions with effects on voters $w_{x,y}$ and $w'_{x,y}$.

In other words, we select the vertices and the edges corresponding to the clique. In effect, the scores of all the candidates is $\alpha + \beta + 1$ (except for d, who receives lower score). So p is among the tied winners.

For the "only if" part, assume that there is a successful set of shift actions and consider the election after applying these shift actions. By construction, we know that edge-selecting voters pick exactly one edge for each pair of distinct colors. Hence the graph induced by these edges contains vertices with h different colors. If this graph contains only h vertices, then this graph must be an h-colored clique (this graph cannot contain fewer than h vertices). For the sake of contradiction, let us assume that this graph contains more than h vertices. Thus there are two selected edges, e_j and $e_{j'}$, incident to two different vertices, $v_i \in e_j$ and $v_{i'} \in e_{j'}$, of the same color. By our construction (and the way vertex-selecting voters work), for at least one of the sets $N(v_i)$ and $N(v_{i'})$ all candidates in the set receive $\alpha + 1$ points from the vertex-selecting voters. However, since both e_j and $e_{j'}$ are selected by the edge-selecting voters, these voters give $\beta + 1$ points to each of the candidates e_j^1, e_j^2, $e_{j'}^1$, and $e_{j'}^2$. Hence, at least one of these candidates receives $\alpha + \beta + 2$ points in total and, by Lemma 1, p is not a winner. This is a contradiction, and so the graph induced by the selected edges must be an h-colored clique. □

References

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

Baumeister, D., Faliszewski, P., Lang, J., & Rothe, J. (2012). Campaigns for lazy voters: Truncated ballots. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '12)*, pp. 577–584. IFAAMAS.

Betzler, N., Bredereck, R., Chen, J., & Niedermeier, R. (2012). Studies in computational aspects of voting—a parameterized complexity perspective. In *The Multivariate Algorithmic Revolution and Beyond*, Vol. 7370 of *LNCS*, pp. 318–363. Springer.

Binkele-Raible, D., Erdélyi, G., Fernau, H., Goldsmith, J., Mattei, N., & Rothe, J. (2014). The complexity of probabilistic lobbying. *Discrete Optimization*, *11*, 1–21.

Boutilier, C., Brafman, R. I., Hoos, C. D. H. H., & Poole, D. (2004). CP-nets: A tool for representing and reasoning with conditional *ceteris paribus* preference statements. *Journal of Artificial Intelligence Research*, *21*, 135–191.

Brandt, F., Harrenstein, P., Kardel, K., & Seedig, H. G. (2013). It only takes a few: On the hardness of voting with a constant number of agents. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '13)*, pp. 375–382. IFAAMAS.

Bredereck, R., Chen, J., Faliszewski, P., Nichterlein, A., & Niedermeier, R. (2014a). Prices matter for the parameterized complexity of shift bribery. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI '14)*, pp. 1398–1404. AAAI Press.

Bredereck, R., Chen, J., Hartung, S., Kratsch, S., Niedermeier, R., Suchý, O., & Woeginger, G. (2014b). A multivariate complexity analysis of lobbying in multiple referenda. *Journal of Artificial Intelligence Research*, *50*, 409–446.

Bredereck, R., Faliszewski, P., Niedermeier, R., Skowron, P., & Talmon, N. (2015a). Elections with few candidates: Prices, weights, and covering problems. In *the Fourth International Conference on Algorithmic Decision Theory (ADT 2015)*, Vol. 9346 of *LNCS*, pp. 414–431. Springer.

Bredereck, R., Faliszewski, P., Niedermeier, R., & Talmon, N. (2015b). Large-scale election campaigns: Combinatorial shift bribery. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'15)*, pp. 67–75.

Bredereck, R., Faliszewski, P., Niedermeier, R., & Talmon, N. (2016). Complexity of shift bribery in committee elections. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI '16)*.

Bulteau, L., Chen, J., Faliszewski, P., Niedermeier, R., & Talmon, N. (2015). Combinatorial voter control in elections. *Theoretical Computer Science*, *589*, 99–120.

Cary, D. (2011). Estimating the margin of victory for instant-runoff voting. Presented at the 2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections.

Chen, J., Faliszewski, P., Niedermeier, R., & Talmon, N. (2015). Elections with few voters: Candidate control can be easy. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI '15)*, pp. 2045–2051.

Christian, R., Fellows, M. R., Rosamond, F. A., & Slinko, A. (2007). On complexity of lobbying in multiple referenda. *Review of Economic Design*, *11*(3), 217–224.

Conitzer, V., Lang, J., & Xia, L. (2009). How hard is it to control sequential elections via the agenda?. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI '10)*, pp. 103–108. AAAI Press.

Dorn, B., & Schlotter, I. (2012). Multivariate complexity analysis of swap bribery. *Algorithmica*, *64*(1), 126–151.

Downey, R. G., & Fellows, M. R. (2013). *Fundamentals of Parameterized Complexity*. Springer.

Elkind, E., & Faliszewski, P. (2010). Approximation algorithms for campaign management. In *Proceedings of the 6th International Workshop on Internet and Network Economics (WINE '10)*, Vol. 6484 of *LNCS*, pp. 473–482. Springer.

Elkind, E., Faliszewski, P., & Slinko, A. (2009). Swap bribery. In *Proceedings of the 2nd International Symposium on Algorithmic Game Theory (SAGT '09)*, Vol. 5814 of *LNCS*, pp. 299–310. Springer.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. (2010). Using complexity to protect elections. *Communications of the ACM*, *53*(11), 74–82.

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. A. (2009a). How hard is bribery in elections?. *Journal of Artificial Intelligence Research*, *35*, 485–532.

Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L. A., & Rothe, J. (2009b). Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, *35*, 275–341.

Faliszewski, P., Reisch, Y., Rothe, J., & Schend, L. (2014). Complexity of manipulation, bribery, and campaign management in Bucklin and Fallback voting. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '14)*, pp. 1357–1358. IFAAMAS.

Faliszewski, P., & Rothe, J. (2015). Control and bribery in voting. In Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.), *Handbook of Computational Social Choice*, chap. 7. Cambridge University Press.

Flum, J., & Grohe, M. (2006). *Parameterized Complexity Theory*. Springer.

Gabow, H. N. (1983). An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC '83)*, pp. 448–456. ACM.

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman.

Hazon, N., Lin, R., & Kraus, S. (2013). How to change a group's collective decision?. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI '13)*, pp. 198–205. AAAI Press.

Hulett, H., Will, T. G., & Woeginger, G. J. (2008). Multigraph realizations of degree sequences: Maximization is easy, minimization is hard. *Operations Research Letters*, *36*(5), 594–596.

Lang, J., & Xia, L. (2015). Voting in combinatorial domains. In Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.), *Handbook of Computational Social Choice*, chap. 9. Cambridge University Press.

Lenstra, H. W. (1983). Integer programming with a fixed number of variables. *Mathematics of Operations Research*, *8*(4), 538–548.

Magrino, T., Rivest, R., Shen, E., & Wagner, D. (2011). Computing the margin of victory in IRV elections. Presented at the 2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections.

Mathieson, L., & Szeider, S. (2012). Editing graphs to satisfy degree constraints: A parameterized approach. *Journal of Computer and System Sciences*, *78*(1), 179–191.

Mattei, N., Goldsmith, J., & Klapper, A. (2012a). On the complexity of bribery and manipulation in tournaments with uncertain information. In *Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference (FLAIRS '12)*, pp. 549–554. AAAI Press.

Mattei, N., Pini, M., Rossi, F., & Venable, K. (2012b). Bribery in voting over combinatorial domains is easy. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '12)*, pp. 1407–1408. IFAAMAS.

Niedermeier, R. (2006). *Invitation to Fixed-Parameter Algorithms*. Oxford University Press.

Obraztsova, S., & Elkind, E. (2011). On the complexity of voting manipulation under randomized tie-breaking. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI '11)*, pp. 319–324. AAAI Press.

Obraztsova, S., Elkind, E., & Hazon, N. (2011). Ties matter: Complexity of voting manipulation revisited. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '11)*, pp. 71–78.

Rajagopalan, S., & Vazirani, V. V. (1998). Primal-dual RNC approximation algorithms for set cover and covering integer programs. *SIAM Journal on Computing*, *28*(2), 525–540.

Reisch, Y., Rothe, J., & Schend, L. (2014). The margin of victory in Schulze, Cup, and Copeland elections: Complexity of the regular and exact variants. In *Proceedings of the Seventh European Starting AI Researcher Symposium (STAIRS-2014)*, pp. 250–259. IOS Press.

Schlotter, I., Faliszewski, P., & Elkind, E. (2011). Campaign management under approval-driven voting rules. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI '11)*, pp. 726–731. AAAI Press.

Xia, L. (2012). Computing the margin of victory for various voting rules. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC '12)*, pp. 982–999. ACM Press.