How Hard is Safe Bribery?

Neel Karia*
Microsoft Research
Bengaluru, India
t-neelkaria@microsoft.com

Faraaz Mallick*
IIT Kharagpur
Kharagpur, India
faraazrm@iitkgp.ac.in

Palash Dey
IIT Kharagpur
Kharagpur, India
palash.dey@cse.iitkgp.ac.in

ABSTRACT

Bribery in an election is one of the well-studied control problems in computational social choice. In this paper, we propose and study the safe bribery problem. Here the goal of the briber is to ask the bribed voters to vote in such a way that the briber never prefers the original winner (of the unbribed election) more than the new winner, even if the bribed voters do not fully follow the briber's advice. Indeed, in many applications of bribery, campaigning for example, the briber often has limited control on whether the bribed voters eventually follow her recommendation and thus it is conceivable that the bribed voters can either partially or fully ignore the briber's recommendation. We provide a comprehensive complexity theoretic landscape of the safe bribery problem for many common voting rules in this paper.

KEYWORDS

Bribery; Voting; Social Choice; Safe Bribery; Shift Bribery; Computational Complexity; Parameterized Hardness; Algorithms

ACM Reference Format:
Neel Karia, Faraaz Mallick, and Palash Dey. 2022. How Hard is Safe Bribery?. In *Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), Online, May 9–13, 2022*, IFAAMAS, 9 pages.

1 INTRODUCTION

Voting has always served as a fundamental tool for aggregating varied preferences in many applications in real-life and artificial intelligence [32, 42]. In a typical voting setting, we have a set of candidates, a set of voters each having a preference over the set of candidates, and a voting rule which decides a winner based on the preferences of the voters. Any such voting scenario is susceptible to various kinds of control attacks – voters or candidates or some other third party may influence the outcome of the election in their way through some unfair means [27]. One of the most well-studied attacks of such type is *bribery,* where an external agent, called a briber, offers monetary rewards to some voters to vote as the briber suggests so that the favourite candidate of the briber wins the election [21–23]. This bribery problem not only models monetary bribing but also other situations like campaigning in an election where the monetary reward corresponds to the amount of time and energy one needs to spend to campaign for some candidate. Depending on how the briber needs to pay the voters to change their votes, various models have been studied. In the $\textsc{\$Bribery}$ problem, each voter has a fixed cost that the briber needs to pay to make the voter cast the vote of the briber's choice [21, 22]. In $\textsc{Swap Bribery}$, the briber has to pay for each swap of consecutive candidates in a voter's preference and the price also depends on the pair of candidates being swapped [16]. In $\textsc{Shift Bribery}$, the briber can only shift her favourite candidate by some positions and the cost depends on the number of positions shifted [4, 5, 33, 39].

To the best of our knowledge, all the existing work on bribery assumes that the bribed voters cast the exact same vote that the briber has asked them. Although this may be a reasonable assumption for some applications, in many other applications of bribery, say campaigning, the briber can never be sure that the voters will eventually cast the vote of the briber's choice. Indeed, it may very well be the case that some subset of voters follows the briber's recommendation exactly, some other subset of voters follows them partially, and the remaining voters complete ignore the briber. Moreover, the briber may not have any knowledge of these sets of voters. Now the situation will be worse for the briber if she prefers the original winner more than the winner of the resulting election, where not all bribed voters cast the votes of the briber's choice. For example, let us consider a plurality election where 10 voters vote for candidate a, 8 voters vote for candidate b, and 4 voters vote for candidate c. Assume that the briber's preference is $c \succ a \succ b$ and the briber bribes 6 voters voting originally for a to vote for c. However, only 3 of the 6 bribed voters eventually follow the briber's recommendation, while the other 3 bribed voters simply ignore the briber. In the resulting election, candidates a and c receive 7 votes each whereas the candidate b wins the election with 8 votes. We observe that the briber would prefer the original winner a over the new winner b. To model this kind of applications more suitably, we propose the *safe bribery* problem.

In the safe bribery problem, the briber has a preference $\succ_B$ which is a complete order over the candidates. Given a preference profile of a set of voters, a cost function for each voter, a favourite candidate c of the briber, and a budget of the briber, the goal of the safe bribery problem is to compute if there exists a subset of voters who can be bribed in such a way that (i) if all the bribed voters follow the briber's recommendation, then c wins, and (ii) for every subset of bribed voters who follows the briber's recommendation exactly, every subset of bribed voters who follows the briber's recommendation partially, and the other bribed voters ignoring the briber's recommendation, the winner of the election is not less preferred than the winner of the "unbribed" election according to $\succ_B$. We study the safe version of the $\textsc{\$Bribery}$ and $\textsc{Shift Bribery}$ problems in this paper. We also study the computational problem, called *is safe,* of deciding if a given bribed profile and a given unbribed profile is safe for the briber with respect to a preference of the briber. We refer the reader to Section 2.2 for the formal definitions of the four problems. Section 3.1 covers some algorithms for the polynomial-time results, Section 3.2 covers the hardness results, and Section 4 deals with the parameterized complexity results.

*Both authors contributed equally to the paper

Proc. of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), P. Faliszewski, V. Mascardi, C. Pelachaud, M.E. Taylor (eds.), May 9–13, 2022, Online.

1.1 Contribution

We provide a comprehensive complexity theoretic landscape of the $Bribery Is Safe, Safe $Bribery, Shift Bribery Is Safe, and Safe Shift Bribery problems. We summarise our main results in Table 1. Other than these results, we also show that all the four problems are polynomial time solvable for every anonymous and efficient voting rule when we have a constant number of candidates [Theorems 3.1 and 3.2]. We also look at safety in Shift Bribery from a parameterized hardness perspective, and summarise our results in Tables 2 and 3.

1.2 Related Work

Faliszewski et al. [21] propose the first bribery problem where the briber's goal is to change the minimum number of preferences to make some candidates win the election. Then they extend their basic model to more sophisticated models, including $Bribery [22, 23]. Elkind et al. [16] extend this model further and study the Swap Bribery problem (where there is a cost associated with every swap of candidates), and its special case, the Shift Bribery problem. Dey et al. [11] show that the bribery problem remains intractable for many common voting rules for an interesting special case which they call frugal bribery. The bribery problem has also been studied in various other preference models, for example, truncated ballots [1], soft constraints [43], approval ballots [45], campaigning in societies [20], CP-nets [12], combinatorial domains [38], iterative elections [39], committee selection [5], probabilistic lobbying [3], local distance restricted bribery [10] etc. Erdelyi et al. [18] study the bribery problem under voting rule uncertainty. Faliszewski et al. [25] study bribery for the simplified Bucklin and the Fallback voting rules. Xia [48], and Kaczmarczyk and Faliszewski [33] study the destructive variant of bribery. Dorn and Schlotter [13] and Bredereck et al. [4] explore the parameterized complexity of various bribery problems. Chen et al. [7] provide novel mechanisms to protect elections from bribery. Knop et al. [36] provide a uniform framework for various control problems. Although most of the bribery problems are intractable, a few of them, Shift Bribery for example, have polynomial time approximation algorithms [15, 35]. Manipulation, a specialization of bribery, is another fundamental attack on election [8]. In the manipulation problem, a set of voters (called manipulators) wants to cast their preferences in such a way that (when tallied with the preferences of other preferences) makes some candidate win the election. Obraztsova and Elkind [40, 41] initiate the study of optimal manipulation in that context.

The concept of safety in electoral control problems has been studied before as well. Slinko and White initiate this line of work by proposing the notion of safety in the context of manipulation and studying the class of social choice functions that are safely manipulable [46, 47]. Hazon and Elkind [30] and Ianovski et al. [31] study computational complexity of safely manipulating popular voting rules.

2 PRELIMINARIES

An election is a pair $(\mathcal{C}, \mathcal{V})$, where $\mathcal{C} = \{c_1, \ldots, c_m\}$ is a set of candidates and $\mathcal{V} = \{v_1, \ldots, v_n\}$ is a set of voters. If not mentioned otherwise, we use n and m to respectively denote the number of voters and candidates. Each voter v_i has a preference order (vote) $\succ_i$, which is a linear order over $\mathcal{C}$. We denote the set of all complete orders over $\mathcal{C}$ by $\mathcal{L}(\mathcal{C})$. We call a list of n preference orders $\{\succ_1, \succ_2, \ldots, \succ_n\} \in \mathcal{L}(\mathcal{C})^n$ an n-voter preference profile. We denote the i^{th} preference order of a preference profile $\mathcal{P}$ as $\succ_i^{\mathcal{P}}$. A map $r : \mathcal{L}(\mathcal{C})^n \to \mathcal{C}$ is called a resolute voting rule (as we assume the unique-winner model); in case of ties, the winner is decided by a *lexicographic* tie-breaking mechanism $\succ_t$, which is some pre-fixed ordering over the candidates. For a set of candidates X, let $\overrightarrow{X}$ be an ordering over them. Then, $\overleftarrow{X}$ denotes the reversed ordering of $\overrightarrow{X}$.

Let $[\ell]$ represent the set of positive natural numbers up to ℓ for any positive integer ℓ. A voting rule r is called *anonymous* if, for every preference profile $(\succ_i)_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ and permutation σ of $[n]$, we have $r((\succ_i)_{i\in[n]}) = r((\succ_{\sigma(i)})_{i\in[n]})$. A voting rule is called *efficient* if the winner can be computed in polynomial time of the input length.

A scoring rule is induced by an m-dimensional vector, $(\alpha_1, \ldots, \alpha_m) \in \mathbb{Z}^m$ with $\alpha_1 \geqslant \alpha_2 \geqslant \ldots \geqslant \alpha_m$ and $\alpha_1 > \alpha_m$. A candidate gets a score of α_i from a voter if she is placed at the i^{th} position in the voter's preference order. The score of a candidate from a voter set is the sum of the scores she receives from each of the voters. If α_i is 1 for $i \in [k]$ and 0 otherwise, we get the k-approval rule. If α_i is 0 for $i \in [m-k]$ and -1 otherwise, we get the k-veto rule. The scoring rules for score vectors $(1, 0, \ldots, 0)$ and $(0, \ldots, 0, -1)$ are called plurality and veto rules respectively. The scoring rule for score vector $(m-1, m-2, \ldots, 1, 0)$ is known as the Borda rule.

The scores for the other voting rules studied in the paper (apart from scoring rules) are defined as follows. Let $nv(a, b)$ be the difference in the number of votes in which a precedes b and the number of votes in which a succeeds b, for $a, b \in \mathcal{C}$. The maximin score of a candidate a is $\min_{b\neq a} nv(a, b)$. The candidate with the maximum maximin score (after tie-breaking) is the winner. Given $\alpha \in [0, 1]$, the Copeland$^\alpha$ score of a candidate a is $|\{b \neq a : nv(a, b) > 0\}| + \alpha \times |\{b \neq a : nv(a, b) = 0\}|$. The candidate with the maximum Copeland$^\alpha$ score (after tie-breaking) is the winner. We will assume α to be zero, if not mentioned separately. A candidate a that has a positive pairwise score $nv(a, b)$ against all other candidates $b \in \mathcal{C}$ is called a Condorcet winner. Rules which select a Condorcet winner as the winner (whenever it exists) are called Condorcet-consistent rules. Copeland and maximin voting rules are Condorcet-consistent. The simplified Bucklin score of a given candidate a is the minimum number k such that more than half of the voters rank a in their top k positions. The candidate with the lowest simplified Bucklin score (after tie-breaking) is the winner and her score is called the Bucklin winning round.

We use $s(a)$ to denote the total score that a candidate $a \in \mathcal{C}$ gets in an election. Similarly, if $Y \subseteq \mathcal{C}$, we use $s(Y)$ to refer to the score of each candidate from Y; e.g. $s(Y) = 5$ means that each candidate from Y has a score of 5. The voting rule under consideration will be clear from the context. Note that we assume that the briber is aware of the votes cast by each voter.

2.1 Parameterized Complexity

A parameterized problem Π is a subset of $\Gamma^* \times \mathbb{N}$, where Γ is a finite alphabet. A central notion is *fixed parameter tractability* (FPT) which means, for a given instance (x, k), solvability in time $f(k) \cdot$

Voting Rule	$Bribery Is Safe	Safe $Bribery	Shift Bribery Is Safe	Safe Shift Bribery
Plurality	P (Theorem 3.3)	P (Theorem 3.4)	P (Theorem 3.3)	P (Theorem 3.4)
k-approval	co-NP-complete (Theorem 3.9)	NP-hard (Corollary 3.1)	P (Corollary 3.3)	P (Corollary 3.3)
Veto	P (Theorem 3.5)	P (Theorem 3.6)	P (Theorem 3.5)	P (Corollary 3.4)
k-veto	co-NP-complete (Corollary 3.5)	NP-hard (Corollary 3.1)	P (Corollary 3.5)	P (Corollary 3.5)
Borda	co-NP-complete (Theorem 3.10)	NP-hard (Corollary 3.1)	co-NP-complete (Theorem 3.11)	NP-hard (Corollary 3.1)
S. Bucklin	co-NP-complete (Theorem 3.12)	NP-hard (Corollary 3.1)	P (Theorem 3.7)	P (Theorem 3.8)
Copeland	co-NP-complete (Theorem 3.13)	NP-hard (Corollary 3.1)	co-NP-complete (Theorem 3.14)	NP-hard (Corollary 3.1)
Maximin	co-NP-complete (Theorem 3.13)	NP-hard (Corollary 3.1)	co-NP-complete (Theorem 3.15)	NP-hard (Corollary 3.1)

Table 1: Complexity Results for Safe Bribery

Voting Rule	#shifts	#bribed voters	#candidates
Borda	FPT (Theorem 4.1)	co-W[1]-hard (Theorem 4.3)	XP (Theorem 3.1)
Copeland	FPT (Theorem 4.1)	co-W[1]-hard (Theorem 4.4)	XP (Theorem 3.1)
Maximin	FPT (Theorem 4.1)	?	XP (Theorem 3.1)

Table 2: Parameterized Complexity Results for Shift Bribery Is Safe

Voting Rule	#shifts	#candidates	#voters	#bribed voters OR budget
Borda	XP (Theorem 4.2)	XP (Theorem 4.2)	W[1]-hard (Corollary 3.2)	W[2]-hard (Corollary 3.2)
Copeland	W[1]-hard (Corollary 3.2)	XP (Theorem 4.2)	W[1]-hard (Corollary 3.2)	W[2]-hard (Corollary 3.2)
Maximin	XP (Theorem 4.2)	XP (Theorem 4.2)	W[1]-hard (Corollary 3.2)	W[2]-hard (Corollary 3.2)

Table 3: Parameterized Complexity Results for Safe Shift Bribery

$p(|x|)$, where f is an arbitrary function of k and p is a polynomial in the input size $|x|$. There exists a hierarchy of complexity classes above FPT, and showing that a parameterized problem is hard for one of these classes is considered evidence that the problem is unlikely to be fixed-parameter tractable. The main classes in this hierarchy are: $\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \cdots \subseteq \mathsf{W}[P] \subseteq \mathsf{XP}$. We now define the notion of parameterized reduction [9].

Definition 2.1. *Let A, B be parameterized problems. We say that A is **fpt-reducible** to B if there exist functions $f, g : \mathbb{N} \to \mathbb{N}$, a constant $\alpha \in \mathbb{N}$ and an algorithm Φ which transforms an instance (x, k) of A into an instance $(x', g(k))$ of B in time $f(k)|x|^{\alpha}$ so that $(x, k) \in A$ if and only if $(x', g(k)) \in B$.*

2.2 Problem Definition

Let r be any voting rule. We first define when a bribed profile is "safe" in the context of $Bribery. Let $\succ_B$ be the preference of the briber. Intuitively speaking, we say that a bribed profile is safe if no candidate preferred less than the original winner (in $\succ_B$), wins the election when a subset of bribed voters does not follow the briber's suggestion but casts their original votes. Formally, we define the notion of safety for $Bribery as follows.

Definition 2.2. *(Safety for $Bribery): Given a voting rule r, a set $\mathcal{C}$ of m candidates, a set $\mathcal{V}$ of n voters, an n-voter profile $\mathcal{P} = (\succ_i^{\mathcal{P}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$, the preferred candidate $c \in \mathcal{C}$, a preference order $\succ_B \in \mathcal{L}(\mathcal{C})$ of the briber, and another profile $\mathcal{Q} = (\succ_i^{\mathcal{Q}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ (representing a bribed profile), we say that $\mathcal{Q}$ is safe for $\mathcal{P}$ with respect to $\succ_B$ if the following conditions hold. Let us define $w = r(\mathcal{P})$ (where $c \succ_B w$), and $\mathcal{V}_b = \{v_i \mid v_i \in \mathcal{V}, \succ_i^{\mathcal{P}} \neq \succ_i^{\mathcal{Q}}\}$. We call the voters in $\mathcal{V}_b$ the bribed voters, and define $\mathcal{V}_u = \mathcal{V} \setminus \mathcal{V}_b$.*

- ▷ ***[Safety]** For every subset $\mathcal{V}'_b \subseteq \mathcal{V}_b$, if we have $x = r((\succ_i^{\mathcal{Q}})_{v_i\in\mathcal{V}'_b}, (\succ_i^{\mathcal{P}})_{v_i\in\mathcal{V}\setminus\mathcal{V}'_b})$, then we have $x \succ_B w$ or $x = w$.*
- ▷ ***[Success]** We have $c = r((\succ_i^{\mathcal{Q}})_{v_i\in\mathcal{V}_b}, (\succ_i^{\mathcal{P}})_{v_i\in\mathcal{V}_u})$.*

We now define the computation problem of finding if a bribed profile is safe.

Definition 2.3. *($Bribery Is Safe): Given a voting rule r, a set $\mathcal{C}$ of m candidates, an n-voter profile $\mathcal{P} = (\succ_i^{\mathcal{P}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ over $\mathcal{C}$,the preferred candidate $c \in \mathcal{C}$, a preference order $\succ_B \in \mathcal{L}(\mathcal{C})$ of the briber, and another profile $\mathcal{Q} = (\succ_i^{\mathcal{Q}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ (representing the bribed profile), compute if $\mathcal{Q}$ is safe for $\mathcal{P}$ with respect to $\succ_B$ for r. We denote any arbitrary instance of $Bribery Is Safe by $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$.*

We next define the computational problem of safely bribing the voters in an election.

Definition 2.4. *(Safe $Bribery): Given a voting rule r, a set $\mathcal{C}$ of m candidates, a set $\mathcal{V}$ of n voters, an n-voter profile $\mathcal{P} = (\succ_i^{\mathcal{P}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ corresponding to $\mathcal{V}$, the preferred candidate $c \in \mathcal{C}$, a preference $\succ_B \in \mathcal{L}(\mathcal{C})$ of the briber, a family $\Pi = (\pi_i)_{i\in[n]} \in \mathbb{N}^n$ of cost functions corresponding to every voter, and a budget $b \in \mathbb{R}$, compute if there exists a set $\mathcal{V}_b \subseteq \mathcal{V}$ of voters along with a profile $\mathcal{Q} = ((\succ_i^{\mathcal{Q}})_{v_i\in\mathcal{V}_b}, (\succ_i^{\mathcal{P}})_{v_i\in\mathcal{V}\setminus\mathcal{V}_b}) \in \mathcal{L}(\mathcal{C})^n$ such that the following conditions hold: (1) $\sum_{v_i\in\mathcal{V}_b} \pi_i \leqslant b$, (2) The profile $\mathcal{Q}$ is safe for $\mathcal{P}$*

with respect to $\succ_B$ for r. An arbitrary instance of Safe $Bribery is denoted by $(\mathcal{C}, \mathcal{P}, c, \succ_B, \Pi, b)$.

We next define the concept of safety for Shift Bribery. Here, the concept of safety is more fine-grained. Suppose a voter is bribed to shift the favourite candidate c of the briber by t positions. However, it is possible that the bribed voter follows the suggestion of the briber only partially and shifts c to left by a lesser number of positions than t. Hence, the briber needs to bribe in this case in such a way that no unfavourable candidate for the briber (compared to the original winner) wins the election even if any subset of voters follow the briber's suggestion only partially. Formally, it is defined as follows.

Definition 2.5. *(Safety for Shift Bribery): Given a voting rule r, a set $\mathcal{C}$ of m candidates, a set of $\mathcal{V}$ of n voters, an n-voter profile $\mathcal{P} = (\succ_i^{\mathcal{P}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ corresponding to $\mathcal{V}$, the preferred candidate $c \in \mathcal{C}$, a preference order $\succ_B \in \mathcal{L}(\mathcal{C})$ of the briber, and a shift vector $s = (s_1, \ldots, s_n) \in \mathbb{N}_0^n$, we say that the shift vector s is safe for $\mathcal{P}$ with respect to $\succ_B$ if the following conditions hold. Let us define $w = r(\mathcal{P})$ (where $c \succ_B w$) and $\mathcal{Q} = \{\succ_i^{\mathcal{Q}} \mid i \in [n], \succ_i^{\mathcal{Q}}$ is obtained from $\succ_i^{\mathcal{P}}$ by shifting c to the left by s_i positions$\}$*

- ▷ ***(Safety)*** *For every shift vector $s' = (s'_1, \ldots, s'_n)$ with $s'_i \leqslant s_i, \forall i \in [n]$, if we have $\mathcal{Q}' = \{\succ_i^{\mathcal{Q}} \mid i \in [n], \succ_i^{\mathcal{Q}}$ is obtained from $\succ_i^{\mathcal{P}}$ by shifting c to the left by s'_i positions$\}$ and $x = r(\mathcal{Q}')$, then we have $x \succ_B w$ or $x = w$.*
- ▷ ***(Success)*** *We have $c = r(\mathcal{Q})$.*

We next define the problem of computing if a Shift Bribery is safe.

Definition 2.6. *(Shift Bribery Is Safe): Given a voting rule r, a set $\mathcal{C}$ of m candidates, an n-voter profile $\mathcal{P} = (\succ_i^{\mathcal{P}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ over $\mathcal{C}$, a distinguished candidate $c \in \mathcal{C}$, a preference order $\succ_B \in \mathcal{L}(\mathcal{C})$ of the briber, and a shift vector $s = (s_1, \ldots, s_n) \in \mathbb{N}_0^n$ (corresponding to a bribing strategy), compute if s is safe for $\mathcal{P}$ with respect to $\succ_B$ for r. We denote any arbitrary instance of Shift Bribery Is Safe by $(\mathcal{C}, \mathcal{P}, c, \succ_B, s)$. Sometimes it will be convenient to define an instance of Shift Bribery Is Safe as $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$ where $\mathcal{Q}$ is obtained by applying s to $\mathcal{P}$.*

We next define the computational problem of bribing the voters in an election, safely and according to the rules of Shift Bribery.

Definition 2.7. *(Safe Shift Bribery): Given a voting rule r, a set $\mathcal{C}$ of m candidates, a set $\mathcal{V}$ of n voters, an n-voter profile $\mathcal{P} = (\succ_i^{\mathcal{P}})_{i\in[n]} \in \mathcal{L}(\mathcal{C})^n$ over $\mathcal{C}$, the preferred candidate $c \in \mathcal{C}$, a preference $\succ_B \in \mathcal{L}(\mathcal{C})$ of the briber, a family $\Pi = (\pi_i : [m-1] \to \mathbb{N})_{i\in[n]}$ of cost functions (where $\pi_i(0) = 0, \forall i \in [n]$) corresponding to every voter, and a budget $b \in \mathbb{R}$, compute if there exists a shift vector $s = (s_1, \ldots, s_n) \in \mathbb{N}_0^n$ such that: (1) $\sum_{v_i\in\mathcal{V}} \pi_i(s_i) \leqslant b$ (2) The shift vector s is safe for $\mathcal{P}$ with respect to $\succ_B$. An arbitrary instance of Safe Shift Bribery is denoted by $(\mathcal{C}, \mathcal{P}, c, \succ_B, \Pi, b)$.*

3 RESULTS

We now present our results. Given an election with its winner w and the preference $\succ_B$ of the briber, we define a set G of "good candidates" as $\{a \in \mathcal{C} : a \succ_B w\} \cup \{w\}$ and the set B of "bad candidates" as $\{a \in \mathcal{C} : w \succ_B a\}$. For Is Safe, let $\mathcal{V}_b$ be the set of bribed voters, and $\mathcal{V}_u$ the rest of the voters.

We begin with showing a connection of Safe Shift Bribery and Safe $Bribery with the classical problems Shift Bribery and $Bribery respectively.

Observation 3.1. *There is a polynomial-time many-to-one reduction from $Bribery to Safe $Bribery and from Shift Bribery to Safe Shift Bribery.*

Let w be the winner of the "unbribed election". To reduce any instance of $Bribery (Shift Bribery respectively) to Safe $Bribery (Safe Shift Bribery respectively), we set the preference $\succ_B$ of the briber to be $c \succ \ldots \succ w$ and keep everything else the same. The reduction is clearly correct and runs in polynomial time.

Many hardness results of Safe $Bribery and Safe Shift Bribery are obtained as useful corollaries.

Corollary 3.1. *If $Bribery (Shift Bribery respectively) is NP-complete for any anonymous and efficient voting rule, Safe $Bribery (Safe Shift Bribery respectively) is NP-hard for that voting rule.*

Since $Bribery is NP-complete for k-approval, k-veto, Borda, simplified Bucklin, Copeland and maximin [6, 21, 23, 26], Safe $Bribery is NP-hard for these voting rules. Similarly, since Shift Bribery is NP-complete for Borda, Copeland and maximin (using results from [17]), Safe Shift Bribery is NP-hard for these voting rules.

Corollary 3.2. *If Shift Bribery is W[k]-hard for any anonymous and efficient voting rule, when parameterized by any parameter, Safe Shift Bribery is also W[k]-hard for that voting rule, when parameterized by the same parameter.*

This is because the reduction in Observation 3.1 is also an fpt-reduction, since the instances of the Shift Bribery and Safe Shift Bribery problems are the same (preserving the parameter), with the only difference being the addition of $\succ_B$ to the Safe Shift Bribery instance, which can be done in polynomial time. Since Shift Bribery is W[2]-hard for Borda, Copeland and maximin when parameterized by the number of bribed voters or the budget [4, 5], Safe Shift Bribery is also W[2]-hard for these voting rules when parameterized by either of these two parameters. Since Shift Bribery is W[1]-hard for Borda, Copeland, and maximin when parameterized by the number of voters, Safe Shift Bribery is also W[1]-hard for these three voting rules with number of bribed voters as the parameter [4]. By a similar reduction, Safe Shift Bribery for Copeland is W[1]-hard with respect to the number of shifts, given that Shift Bribery for Copeland is W[1]-hard with respect to the number of shifts [4].

3.1 Algorithmic Results

We present here our algorithmic results. We show that all our four problems are polynomial-time solvable for every anonymous and efficient voting rule when we have a constant number of candidates. Our results on the scoring rules follow via an algorithm that uses min-cost flow problem as a crucial subroutine, in a similar manner as in [19]. In the interest of space, we omit a few proofs. For the complete proofs, please refer to [34].

Theorem 3.1. *When we have a constant number of candidates, both $Bribery Is Safe and Shift Bribery Is Safe are in* P *for every anonymous and efficient voting rule. That is, both $Bribery Is Safe and Shift Bribery Is Safe belong to* XP *with respect to the number of candidates as the parameter.*

Proof Sketch. Let $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$ and $(\mathcal{C}, \mathcal{P}, c, \succ_B, s)$ be any instances of $Bribery Is Safe and Shift Bribery Is Safe respectively. If $r(\mathcal{Q})$ is not c, we output no since the bribery is not successful in this case. The number of possible anonymous preference profiles is $\binom{m!+n-1}{m!-1} = \mathcal{O}\left((n+m)^m\right) = \mathcal{O}\left(n^{\mathcal{O}(1)}\right)$ when we have $m = \mathcal{O}(1)$. Let $\mathcal{R}$ be any anonymous preference profile such that $r(\mathcal{P}) \succ_B r(\mathcal{R})$; if no such $\mathcal{R}$ exists, we output yes. We now construct a flow network $\mathcal{G}_{\mathcal{R}}(V, E, W)$ to verify if one can obtain the profile $\mathcal{R}$ from $\mathcal{P}$ using some bribed voters ignoring the briber's suggestion fully (or partially for shift safe bribery).

$$V = \{a_i, b_i \mid i \in [n]\} \cup \{s, t\}$$
$$E = \{(s, a_i) \mid i \in [n]\} \cup \{(b_i, t) \mid i \in [n]\} \cup F$$

We now describe the set F of edges. There is an edge (a_i, b_j) in F if for $Bribery Is Safe, $\succ_i^{\mathcal{Q}} = \succ_j^{\mathcal{R}}$ or $\succ_i^{\mathcal{P}} = \succ_j^{\mathcal{R}}$ and for Shift Bribery Is Safe if $\succ_j^{\mathcal{R}}$ can be obtained from $\succ_i^{\mathcal{P}}$ by moving c left by at most s_i positions. We finally define the capacity of every edge to be 1. It is easy to check that one can obtain the profile $\mathcal{R}$ from $\mathcal{P}$ considering some bribed voters who ignore the briber's suggestion fully (or partially for shift safe bribery) if and only if there is an $s-t$ flow of value n. We output yes if there is no $\mathcal{R}$ such that there exists an $s-t$ flow of value n. Otherwise, we output no. Since maximum $s-t$ flow can be computed in polynomial time, our algorithm runs in $\mathcal{O}\left((n+m)^m\right)\text{poly}(m, n)$ time.

Theorem 3.2. *When we have a constant number of candidates, both Safe $Bribery and Safe Shift Bribery are in* P *for every anonymous and efficient voting rule. That is, both Safe $Bribery and Safe Shift Bribery belong to* XP *with respect to the number of candidates as the parameter.*

The algorithm is again based on a flow network construction and runs in $\mathcal{O}\left((n+m)^{2m}\right)\text{poly}(m, n)$ time.

We next present our results for specific voting rules, starting with the plurality voting rule.

Theorem 3.3. *For plurality, both $Bribery Is Safe and Shift Bribery Is Safe are in* P.

Proof Sketch. Let $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$ be any instance of $Bribery Is Safe for plurality. For every bad candidate $b \in B$, we define a set $W_b = \{i \mid i \in [n], b \text{ is the top candidate in } \succ_i^{\mathcal{P}}\}$. Let $|W_b| = n_b$. If $r(\mathcal{Q})$ is not c, we output no, as the bribed profile is not successful. Else, we try to find a preference profile, where a bad candidate can win, subject to the constraints of $Bribery Is Safe. To model this, we construct a flow network for every $b \in B$, named $\mathcal{G}_b = (V, E, W)$, to check if it is possible for b to win the election by making some bribed voters not to fully follow the briber's suggestion.

$$V = \{x_i \mid i \in [n]\} \cup \{y_a \mid a \in \mathcal{C}\} \cup \{s, t\}$$
$$E = \{(s, x_i) \mid i \in [n]\} \cup \{(y_a, t) \mid a \in \mathcal{C}\} \cup F \cup \{(x_i, y_b) \mid i \in W_b\}$$

The capacities W are as follows. For every $i \in [n] \setminus W_b$, we have an edge $(x_i, y_a) \in F$ if a is the top-ranked candidate in $\succ_i^{\mathcal{Q}}$ or $\succ_i^{\mathcal{P}}$. We define the capacity of edge (y_a, t) to be $(n_b - 1)$ for every $a \in \mathcal{C} \setminus \{b\}$ such that a is preferred over b in the tie-breaking rule and n_b for every $a \in \mathcal{C} \setminus \{b\}$ such that b is preferred over a in the tie-breaking rule. The capacity of the edge (y_b, t) is n_b. The capacity of all other edges is 1. It can be shown that b can be made winner if and only if there is an $s-t$ flow of value n in $\mathcal{G}_b$.

If for every bad candidate $b \in B$, the value of maximum $s-t$ flow is less than n, then we output yes. Otherwise, we output no (as in this case, no bad candidate can ever win). Since maximum $s-t$ flow can be computed in $\mathcal{O}\left((m+n)n^2\right)$ using Edmonds-Karp algorithm [14], and we run at most m instances of it, our algorithm runs in $\mathcal{O}\left(m(m+n)n^2\right)$.

Any instance of Shift Bribery Is Safe for plurality $(\mathcal{C}, \mathcal{P}, c, \succ_B, s)$ can be mapped to a $Bribery Is Safe instance by considering the fact that if c appears in the top $s_i + 1$ positions in $\succ_i^{\mathcal{P}}$, then c should be placed at the top position in $\succ_i^{\mathcal{Q}}$, otherwise $\succ_i^{\mathcal{Q}} = \succ_i^{\mathcal{P}}$. This allows us to use the above construction to solve an Shift Bribery Is Safe instance of plurality in $\mathcal{O}\left(m(m+n)n^2\right)$ time.

We next show that Safe $Bribery and Safe Shift Bribery are polynomial-time solvable for the plurality and the voting rule.

Theorem 3.4. *For plurality, both Safe $Bribery and Safe Shift Bribery are in* P.

Proof Sketch. Let $(\mathcal{C}, \mathcal{P}, c, \succ_B, \Pi, b)$ be an arbitrary instance of Safe $Bribery for plurality. Let the plurality winner according to $\mathcal{P}$ be w. We may assume without loss of generality that the briber asks all the bribed voters to have c as their most preferred candidate. G and B are as defined in Theorem 3.3. Let $s(b)$ denote the initial score of candidate b. $b_{max} = \max\{s(b) \mid w \succ_t b\}$, according to the tie breaking rule, $\succ_t$. Let $X = \{a \in G \mid s(a) \geqslant b_{max}, a \succ_t b \text{ or } s(a) > b_{max}\}$. Initially X is non-empty since $w \in X$. Let λ be the score of c in the constructed bribed profile. For all candidates $a \in \mathcal{C}$, let us define β_a to be λ if $c \succ_t a$ and β_a to be $\lambda - 1$ if $a \succ_t c$. For every $x \in X$ and every final score λ of c which is in the range $s(c)$ to n, we construct a flow network $\mathcal{G}_{x,\lambda}(V, E, C, D, W)$, defining γ_x to be b_{max} if $x \succ_t b$, else $(b_{max} + 1)$. Here D is the set of edge demands, w is the set of edge capacities, and C is the set of edge costs. The construction is as follows:

$$V = \{s, t\} \cup \{u_i \mid i \in [n]\} \cup \{y_a \mid a \in \mathcal{C}\}$$
$$E = \{(s, u_i) \mid i \in [n]\} \cup \{(y_a, t) \mid a \in \mathcal{C}\} \cup$$
$$\{(u_i, y_a) \mid i \in [n], a = c \text{ or } \succ_i^{\mathcal{P}} = a \succ \ldots\}$$

Let the capacities of the edges be as follows. For every candidate $a \in \mathcal{C}$, the edges (y_a, t) have a capacity of β_a, and the rest of the edges have capacity 1 each. Let the costs of the edges be defined as follows. The edges (u_i, y_c) have cost π_i, only if c does not appear at the top of $\succ_i^{\mathcal{P}}$. The demands (lower bound of flow) on the edges are as follows. The edge (y_c, t) has a demand of λ, the edge (y_x, t) has a demand of γ_x and the rest of the edges have a demand of 0 each. If there is an $s-t$ flow of value n and cost at most b for some $\mathcal{G}_{x,\lambda}$, then we output yes, and the corresponding edge flow values help us construct a safe and successful bribed profile. Otherwise we output no. Safe Shift Bribery can be reduced to this problem by defining the price of each voter π_i to be the cost required to

shift c to the top in their ordering. This algorithm is polynomial-time solvable, by running $\mathcal{O}(mn)$ iterations of the out-of-kilter algorithm [28].

Corollary 3.3. *For k-approval, both Shift Bribery Is Safe and Safe Shift Bribery are in P, for every integer $k \in [2, m-1]$.*

A bribery instance of k-approval is equivalent to a corresponding bribery instance of plurality, where the cost of moving c to the top in plurality is equal to the cost of moving c to any of the top k positions in k-approval (for every voter). The rest follows from Theorem 3.3 and Theorem 3.4.

Next, we show some polynomial-time results for veto.

Theorem 3.5. *For veto, both $Bribery Is Safe and Shift Bribery Is Safe are in P and can be solved in polynomial time.*

The algorithm uses a similar flow network construction and the time complexity is $\mathcal{O}\left(m(m+n)n^2\right)$.

Theorem 3.6. *For veto, Safe $Bribery is in P.*

The cheapest Safe $Bribery for veto is obtained by using a simple greedy algorithm in $\mathcal{O}\left(n^2 m \log(n)\log(m)\right)$.

Corollary 3.4. *For veto, Safe Shift Bribery is in P.*

The result follows from Corollary 3.3.

Next, we take a look at the greedy algorithm for solving Shift Bribery Is Safe for simplified Bucklin. Although for simplified Bucklin both Shift Bribery Is Safe and Safe Shift Bribery are polynomial-time solvable, their $Bribery counterparts are not.

Theorem 3.7. *For simplified Bucklin, Shift Bribery Is Safe is in P.*

Proof Sketch. We describe a greedy algorithm to solve Shift Bribery Is Safe for simplified Bucklin. Consider $(\mathcal{C}, \mathcal{P}, c, \succ_B, s)$ to be an instance of Shift Bribery Is Safe for simplified Bucklin, with $|\mathcal{C}| = m$. Let the winning candidate according to $\mathcal{P}$ be $w \in \mathcal{C}$ and $w \neq c$.

Let $\mathcal{V}$ be the set of voters. Let ℓ be the simplified Bucklin winning round according to $\mathcal{P}$. It is clear that the winning round for any candidate $x \in \mathcal{C} \setminus \{c\}$ according to $\mathcal{Q}$ is no smaller than ℓ. Particularly, the winning round for w is either ℓ or $\ell+1$ according to $\mathcal{Q}$ (as explained in [44]). Let $sc_\ell(\mathcal{P}, a)$ denote the number of votes received by $a \in \mathcal{C}$, till the ℓ^{th} round according to $\mathcal{P}$.

$$\text{Let } \succ_t = c \succ w \succ \overrightarrow{\mathcal{C}_1} \succ \overrightarrow{\mathcal{C}_2}$$
$$\text{Let } \succ_B = c \succ \overrightarrow{\mathcal{C}_1} \succ w \succ \overrightarrow{\mathcal{C}_2}, \text{where}$$
$$\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \{c, w\}$$

Note that the above mentioned tie-breaking rule is assumed to simplify the proof. Clearly $\mathcal{C}_2$ is the set of *bad* candidates. Let $p_i(\mathcal{X}, a)$ denote the position of candidate a in the i^{th} vote of some profile $\mathcal{X}$. If $r(\mathcal{Q}) \neq c$, we return no, as the bribery is not successful. Otherwise, we use the following algorithm for checking safety of Shift Bribery Is Safe in simplified Bucklin.

Let B' be the complete subset of bad candidates each having number of votes greater than $\lfloor n/2 \rfloor$ at the ℓ^{th} level. These are the only candidates who can cause the bribery to be unsafe. Let $b' \in B'$ be a bad candidate who beats all candidates in $B' \setminus \{b'\}$ in tie-breaking. This is the most "powerful" bad candidate, who would be the first bad candidate to win. Consider a preference profile $\mathcal{R}$ and initialise it to $\mathcal{P}$. Now for every voter $v_i \in \mathcal{V}$, we do the following iteratively:

- ▷ If $\exists a \in \mathcal{C} \setminus \mathcal{C}_2$ such that $p_i(\mathcal{R}, a) = \ell$ and $sc_\ell(\mathcal{R}, a) > \lfloor n/2 \rfloor$.
  - If $0 < p_i(\mathcal{R}, c) - p_i(\mathcal{R}, a) \leqslant s_i$, we create a new vote $\succ_i^{\mathcal{R}_b}$, by shifting c up in $\succ_i^{\mathcal{R}}$, such that $p_i(\mathcal{R}_b, a) - p_i(\mathcal{R}_b, c) = 1$. We then assign $\succ_i^{\mathcal{R}} = \succ_i^{\mathcal{R}_b}$. If c wins for $\mathcal{R}$, then Shift Bribery Is Safe is a yes instance; terminate.
  - Else, we keep $\mathcal{R}$ unchanged.
- ▷ Else, Shift Bribery Is Safe is a no instance; terminate.

The algorithm runs in $\mathcal{O}(n)$ time.

Theorem 3.8. *For the simplified Bucklin voting rule, Safe Shift Bribery is in P, assuming a monotonous price function.*

The proof involves 2 cases, one of which uses a flow network, and the other uses a dynamic programming algorithm from [44].

3.2 Hardness Results

We now present our hardness results. We use the Exact Cover by 3-Sets problem which is known to be NP-complete [29], in many of our hardness proofs.

Definition 3.1 (Exact Cover by 3-Sets). *Given a universe $\mathcal{U} = \{u_i \mid i \in 3t\}$ of $3t$ elements and a collection $\mathcal{S} = \{S_i \mid i \in [m]\}$ of subsets of $\mathcal{U}$, where $|S_i| = 3$ for each $i \in [m]$, compute if there exists a set $I \subseteq [m]$ such that $\forall\, i, j \in I$ and $i \neq j$, $S_i \cap S_j = \emptyset$ and $\bigcup_{i \in I} S_i = \mathcal{U}$.*

We show that $Bribery Is Safe is co-NP-complete for the k-approval voting rule for every constant $k \geqslant 3$.

Theorem 3.9. *For k-approval, $Bribery Is Safe is co-NP-complete, for $k \geqslant 3$.*

Proof. To see that $Bribery Is Safe belongs to co-NP, any no instance $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$ can be verified either from the fact that the k-approval winner in $\mathcal{Q}$ is not c or from the existence of a profile $\mathcal{R} = (R_i)_{i \in [n]}$ such that (i) $R_i = \succ_i^{\mathcal{P}}$ or $R_i = \succ_i^{\mathcal{Q}}$ for every $i \in [n]$ and the k-approval winner in $\mathcal{R}$ is less preferred in $\succ_B$ than the k-approval winner in $\mathcal{P}$. To prove co-NP-hardness, we exhibit a reduction from Exact Cover by 3-Sets to $Bribery Is Safe such that the Exact Cover by 3-Sets instance is a yes instance if and only if the $Bribery Is Safe instance is a no instance.

Let $(\mathcal{U} = \{u_1, \ldots, u_{3t}\}, \mathcal{S} = \{S_1, \ldots, S_m\})$ be any instance of Exact Cover by 3-Sets. Without loss of generality, we can assume that $m > t$ by duplicating the sets in $\mathcal{S}$. We construct an instance $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$ of $Bribery Is Safe for k-approval as follows.

$$\mathcal{C} = \mathcal{U} \cup \{x, w\} \cup \mathcal{D} \text{ where}$$
$$\mathcal{D} = \uplus_{i \in [m]} \mathcal{D}_i^1 \uplus_{i \in [m]} \mathcal{D}_i^2 \text{ where } |\mathcal{D}_i^1| = k-3, |\mathcal{D}_i^2| = k-1$$
$$\succ_t = w \succ c \succ \overrightarrow{\mathcal{U} \setminus \{c\}} \succ x \succ \overrightarrow{\mathcal{D}}$$
$$\succ_B = c \succ \overrightarrow{\mathcal{U} \setminus \{c\}} \succ \overrightarrow{\mathcal{D}} \succ w \succ x$$

Here $\mathcal{D}$ is a set of dummy candidates, who cannot win, w is the winner in the "unbribed" preference profile, and c is the winner

in the bribed preference profile. Using Lemma 4.2 from [2], we construct a set of "unbribed voters" and their votes such that we have $s_{\mathcal{V}_u}(u_j) = s_{\mathcal{V}_u}(x) - 2$ for every $j \in [3t]$ and $s_{\mathcal{V}_u}(w) = s_{\mathcal{V}_u}(x) - (m - t + 1)$. We now describe the set $\mathcal{V}_b$ of bribed voters, and their bribed and original votes. For each $i \in [m]$, we have a bribed voter $v_i \in \mathcal{V}_b$, with original vote

$$\succ_i^{\mathcal{P}} = w \succ \overrightarrow{\mathcal{D}_i^2} \succ \overrightarrow{\mathcal{C} \setminus (\{w\} \cup \mathcal{D}_i^2)}$$

and bribed vote

$$\succ_i^{\mathcal{Q}} = \overrightarrow{S_i} \succ \overrightarrow{\mathcal{D}_i^1} \succ \overrightarrow{\mathcal{C} \setminus (S_i \cup \mathcal{D}_i^1)}$$

This finishes the description of the reduced $\textsc{Bribery Is Safe}$ instance. We claim that the $\textsc{Bribery Is Safe}$ instance for k-approval is a no instance if and only if the corresponding instance of Exact Cover by 3-Sets is a yes instance.

$\Longrightarrow$: Suppose the $\textsc{Bribery Is Safe}$ instance is a no instance. We observe that we have only one bad candidate, namely x. Hence, there exists a subset $Y \subseteq \mathcal{V}_b$ such that, if $\mathcal{R}$ is the profile where voters in Y vote according to $\mathcal{Q}$ and every other voter votes as the "unbribed" instance, then the k-approval winner of $\mathcal{R}$ is x. We observe that the k-approval score of x in $\mathcal{R}$ is $s_{\mathcal{V}_u}(x)$. We claim that $|Y| = t$. We have $|Y| \leqslant t$, otherwise there exists some $u_j \in \mathcal{C}$ whose score in $\mathcal{R}$ is at least $s_{\mathcal{V}_u}(u_j)$. However, this contradicts our assumption that x is the k-approval winner in $\mathcal{R}$. Also, $|Y| \geqslant t$, otherwise the score of w in $\mathcal{R}$ is at least $s_{\mathcal{V}_u}(w) + m - t + 1$. However, this contradicts our assumption that x is the k-approval winner in $\mathcal{R}$. Hence, we have $|Y| = t$. Moreover, for x to win, the collection $\{S_i : i \in [m], \succ_i^{\mathcal{Q}} \in Y\}$ of sets forms an exact cover of $\mathcal{U}$ as this is the only case which makes the k-approval score of w and u_j for every $j \in [3t]$ less than the k-approval score of x in $\mathcal{R}$. Therefore, Exact Cover by 3-Sets is a yes instance.

$\Longleftarrow$: Suppose the Exact Cover by 3-Sets instance is a yes instance. Let $X \subseteq \mathcal{S}$ be an exact cover of $\mathcal{U}$. Let us consider the preference profile $\mathcal{R}$ where only bribed-voters in $\{v_i \in \mathcal{V}_b : S_i \in X\}$ vote according to $\mathcal{Q}$ and others vote according to $\mathcal{P}$. The k-approval score of the bad candidate x is $s_{\mathcal{V}_u}(x)$, every candidate in $\{u_j : j \in [3t]\} \cup \{w\}$ is $s_{\mathcal{V}_u}(x) - 1$, and every candidate in $\mathcal{D}$ is at most 1 in $\mathcal{R}$. Hence, x is the k-approval winner in $\mathcal{R}$. Thus, the $\textsc{Bribery Is Safe}$ instance is a no instance. □

Corollary 3.5. *For k-veto, Shift Bribery Is Safe and Safe Shift Bribery are in* P*, for* $k > 1$*; $\textsc{Bribery Is Safe}$ is* co-NP*-complete, for* $k \geqslant 3$.

This follows from Theorem 3.9 and Corollary 3.3.

To get the hardness result for $\textsc{Bribery Is Safe}$ for Borda, we use the Exact Cover by (3,4)-Sets problem, which is known to be NP-complete [24].

Definition 3.2 (Exact Cover by (3,4)-Sets). *Given a universe* $\mathcal{U} = \{u_1, u_2, \ldots, u_{4m/3}\}$ *of* $4m/3$ *elements and a collection* $\mathcal{S} = \{S_i \mid i \in [m]\}$ *of subsets of* $\mathcal{U}$*, where* $|S_i| = 4$ *for each* $i \in [m]$ *and where each* $u_i \in \mathcal{U}$ *is in exactly* 3 *sets* S_j, S_k, S_l *for some* $j, k, l \in [m]^3$*, compute if there exists a set* $I \subseteq [m]$ *such that* $\forall i, j \in I, i \neq j, S_i \cap S_j = \emptyset$ *and* $\bigcup_{i \in I} S_i = \mathcal{U}$.

Theorem 3.10. *For Borda, $\textsc{Bribery Is Safe}$ is* co-NP*-complete.*

Proof Sketch. Firstly $\textsc{Bribery Is Safe}$ for Borda belongs to co-NP, provable in a similar fashion as in Theorem 3.9. To prove co-NP-hardness, we demonstrate a reduction from Exact Cover by (3,4)-Sets to $\textsc{Bribery Is Safe}$ such that the Exact Cover by (3,4)-Sets instance is a yes instance if and only if the $\textsc{Bribery Is Safe}$ instance is a no instance.

Let $(\mathcal{U} = \{u_1, u_2, \ldots, u_{4m/3}\}, \mathcal{S} = \{S_i \mid i \in [m]\})$ be an instance of Exact Cover by (3,4)-Sets. Let $(\mathcal{C}, \mathcal{P}, c, \succ_B, \mathcal{Q})$ be an instance of $\textsc{Bribery Is Safe}$ for Borda. Let $\mathcal{C} = \mathcal{U} \cup \{w, x\} \cup \mathcal{D}$, where $c = u_j$ for some $j \in [4m/3]$ is the winner according to $\mathcal{Q}$, and w is the winner according to $\mathcal{P}$. $\mathcal{D}$ is a set of $\mathcal{O}(m^2)$ dummy candidates (who can never win). These candidates are added because the proof requires that $|\mathcal{C}| = \beta m^2 + \gamma$, where $\beta \geqslant 5/3$ and $\gamma \geqslant 15$.

Let $\succ_B$ be $c \succ \overrightarrow{\mathcal{U} \setminus \{c\}} \succ \overrightarrow{\mathcal{D}} \succ w \succ x$. Here x is the only bad candidate. Let $\succ_t$ be $w \succ c \succ \overrightarrow{\mathcal{U} \setminus \{c\}} \succ x \succ \overrightarrow{\mathcal{D}}$. Consider $\mathcal{V}$ as the set of voters. Let $\mathcal{V}_b$ be the set of m bribed voters (each corresponding to an $S_i \in \mathcal{S}$), and $\mathcal{V}_u$ be the rest of the voters. Using Lemma 4.2 from [2], we can construct the election such that if we consider only the votes in $\mathcal{V}_u$, the scores of the candidates are in the order $s_{\mathcal{V}_u}(x) > s_{\mathcal{V}_u}(\mathcal{U}) > s_{\mathcal{V}_u}(w) > s_{\mathcal{V}_u}(\mathcal{D})$, with $s_{\mathcal{V}_u}(x) - s_{\mathcal{V}_u}(w) = (|\mathcal{C}| - 1)(2m/3 + 1) + (m/3 - 1)$ and $s_{\mathcal{V}_u}(x) - s_{\mathcal{V}_u}(u_i) = 2\beta m^2 + 2\gamma - 12$. For each $i \in [m]$, we have a bribed voter $v_i \in \mathcal{V}_b$, with original vote

$$\succ_i^{\mathcal{P}} = w \succ \overrightarrow{\mathcal{D}} \succ \overrightarrow{\mathcal{U} \setminus S_i} \succ \overrightarrow{S_i} \succ x$$

and bribed vote

$$\succ_i^{\mathcal{Q}} = \overrightarrow{S_i} \succ \overrightarrow{\mathcal{D}} \succ \overrightarrow{\mathcal{U} \setminus S_i} \succ w \succ x$$

It can be shown that an instance of $\textsc{Bribery Is Safe}$ for Borda is a no instance if and only if the corresponding instance of Exact Cover by (3,4)-Sets is a yes instance.

It turns out (by reducing from the Exact Cover by 3-Sets problem) that for Borda, Shift Bribery Is Safe is also co-NP-complete.

Theorem 3.11. *For Borda, Shift Bribery Is Safe is* co-NP*-complete.*

Another reduction from Exact Cover by 3-Sets can be used to show that for simplified Bucklin, $\textsc{Bribery Is Safe}$ is co-NP-complete.

Theorem 3.12. *For simplified Bucklin, $\textsc{Bribery Is Safe}$ is* co-NP*-complete.*

Next, we present the hardness results for some tournament-based rules. For $\textsc{Bribery Is Safe}$ we obtain a generalized hardness result which is applicable to any Condorcet-consistent voting rule. All the three results below are obtained using separate reductions from the Exact Cover by 3-Sets problem.

Theorem 3.13. *For any Condorcet-consistent voting rule, $\textsc{Bribery Is Safe}$ is* co-NP*-complete.*

Theorem 3.14. *For Copeland, Shift Bribery Is Safe is* co-NP*-complete.*

Theorem 3.15. *For maximin, Shift Bribery Is Safe is* co-NP*-complete.*

Next we show the parameterized hardness results for Shift Bribery Is Safe and Safe Shift Bribery.

4 PARAMETERIZED COMPLEXITY RESULTS

We observe that for each of Copeland, Borda and maximin, Shift Bribery Is Safe is fixed parameter tractable when parameterized by the number of shifts.

Theorem 4.1. *For all anonymous and efficient voting rules, Shift Bribery Is Safe parameterized by the number of shifts is fixed parameter tractable with complexity* $\mathcal{O}\left((t^t)poly(m,n)\right)$.

We next see that for Copeland, Borda and maximin, Safe Shift Bribery is in XP when parameterized by the number of shifts. But for Copeland, we have an added result from Corollary 3.2, that it is W[1]-hard with number of shifts as the parameter.

Theorem 4.2. *For all anonymous and efficient voting rules, Safe Shift Bribery parameterized by the number of shifts is in XP with complexity* $\mathcal{O}\left((n+t)^t(t^t)poly(m,n)\right)$.

To show W[k]-hardness, it is enough to give a parameterized reduction from a known hard problem. Our parameterized hardness proofs for Shift Bribery Is Safe, considering the number of bribed voters as a parameter rely on reduction from the W[1]-hard problem Multicoloured Independent Set.

Definition 4.1 (Multicoloured Independent Set). *Consider a graph* $\mathcal{G}=(V,E)$ *where each vertex has one of* h *colours, and compute whether there are* h *vertices of pairwise-distinct colours such that no two of them are connected by an edge.*

The above problem can be proved to be W[1]-hard, by reducing it from a variant of the Multicoloured Clique problem [37].

In the following theorem, we prove that Shift Bribery Is Safe for Borda is co-W[1]-hard when parameterized by the number of bribed voters. The basic structure of this proof is inspired by [5].

Theorem 4.3. *For Borda, Shift Bribery Is Safe is* co-W[1]*-hard, when parameterized by the number of bribed voters.*

Proof Sketch. Let $(\mathcal{C},\mathcal{P},c,\succ_B,\mathcal{Q})$ be an instance of Shift Bribery Is Safe for Borda. We give a parameterized reduction from the W[1]-hard Multicoloured Independent Set problem. Given a graph $\mathcal{G}=(V(\mathcal{G}),E(\mathcal{G}))$ where each vertex has one of h colours. Let $(\mathcal{G},h)$ be our input instance. Without loss of generality, we assume that the number of vertices of each colour is the same and that there are no edges between vertices of the same colour. We write $V(\mathcal{G})$ to denote the set of $\mathcal{G}$'s vertices, and $E(\mathcal{G})$ to denote the set of $\mathcal{G}$'s edges. Further, for every colour $i\in[h]$, we write $V^{(i)}=\{v_1^{(i)},\dots,v_q^{(i)}\}$ to denote the set of vertices of colour i. For each vertex v, we write $E(v)$ to denote the set of edges incident to v. For each vertex v, we write $\delta(v)$ to denote its degree, i.e., $\delta(v)=|E(v)|$ and we let $\Delta=\max_{u\in V(\mathcal{G})}\delta(u)$ be the highest degree of a vertex $\mathcal{G}$. We form an instance of Shift Bribery Is Safe for Borda as follows. We let the candidate set be $\mathcal{C}=\{c,x\}\cup V(\mathcal{G})\cup E(\mathcal{G})\cup F(\mathcal{G})\cup D'\cup D''$, where $F(\mathcal{G})$, D', and D'' are sets of special dummy candidates. We let D' and D'' have a cardinality of $B+1$ each, where $B=h(q+(q-1)\Delta)$. For each vertex v, we let $F(v)$ be a set of $\Delta-\delta(v)$ dummy candidates, and we let $F(\mathcal{G})=\bigcup_{v\in V(\mathcal{G})}F(v)$. We set $F(-i)=\bigcup_{v\in V(i'),i'\neq i}F(v)$. Let $w\in V(\mathcal{G})\cup E(\mathcal{G})$ be the winner in $\mathcal{P}$. For each vertex v, we define the partial preference order $\overrightarrow{S(v)}$:

$$\overrightarrow{S(v)}: v\succ \overrightarrow{E(v)}\succ \overrightarrow{F(v)}$$

For each colour i, we define $\overrightarrow{R(i)}$ to be a partial preference order that ranks first all members of D', then all vertex candidates of colours other than i, then all edge candidates corresponding to edges that are not incident to a vertex of colour i, then all dummy vertices from $F(-i)$, and finally all candidates from D''. Let the briber's preference order, $\succ_B$, be: $c\succ \overrightarrow{\mathcal{C}\setminus\{c,x,w\}}\succ w\succ x$, and let the tie-breaking rule, $\succ_t$, be $c\succ w\succ \overrightarrow{\mathcal{C}\setminus\{c,w,x\}}\succ x$. We therefore let x be the only bad candidate.

Let $\mathcal{V}_b$ be the set of bribed voters. We define two sets of h voters each, $\mathcal{V}_{b_1}$ and $\mathcal{V}_{b_2}$ such that $\mathcal{V}_b=\mathcal{V}_{b_1}\uplus\mathcal{V}_{b_2}$. For each $i\in[h]$, we have a voter $v_{i_1}\in\mathcal{V}_{b_1}$, whose preferences in $\mathcal{P}$ and $\mathcal{Q}$ are:

$$\succ^{\mathcal{P}}_{i_1}=\overrightarrow{S(v_1^{(i)})}\succ\overrightarrow{S(v_2^{(i)})}\succ\cdots\succ\overrightarrow{S(v_q^{(i)})}\succ c\succ x\succ\overrightarrow{R(i)}$$

$$\succ^{\mathcal{Q}}_{i_1}=c\succ\overrightarrow{S(v_1^{(i)})}\succ\overrightarrow{S(v_2^{(i)})}\succ\cdots\succ\overrightarrow{S(v_q^{(i)})}\succ x\succ\overrightarrow{R(i)}$$

Similarly for each $i\in[h]$ we have a voter $v_{i_2}\in\mathcal{V}_{b_2}$, whose preferences in $\mathcal{P}$ and $\mathcal{Q}$ are:

$$\succ^{\mathcal{P}}_{i_2}=\overleftarrow{S(v_q^{(i)})}\succ\overleftarrow{S(v_{q-1}^{(i)})}\succ\cdots\succ\overleftarrow{S(v_1^{(i)})}\succ c\succ x\succ\overrightarrow{R(i)}$$

$$\succ^{\mathcal{Q}}_{i_2}=c\succ\overleftarrow{S(v_q^{(i)})}\succ\overleftarrow{S(v_{q-1}^{(i)})}\succ\cdots\succ\overleftarrow{S(v_1^{(i)})}\succ x\succ\overrightarrow{R(i)}$$

Let $\mathcal{V}_u$ be the voters who were not bribed. They are of the following types. There are h voters ($\forall i\in[h]$), each of types (i)-1 and (i)-2;

(i)-1: $\overleftarrow{R(i)}\succ x\succ c\succ\overleftarrow{S(v_q^{(i)})}\succ\cdots\succ\overleftarrow{S(v_2^{(i)})}\succ\overleftarrow{S(v_1^{(i)})}$

(i)-2: $\overleftarrow{R(i)}\succ x\succ c\succ\overrightarrow{S(v_1^{(i)})}\succ\cdots\succ\overrightarrow{S(v_{q-1}^{(i)})}\succ\overrightarrow{S(v_q^{(i)})}$

There is 1 voter, each of type (ii)-1 and (ii)-2;

(ii)-1: $\overrightarrow{F(\mathcal{G})}\succ\overrightarrow{V(\mathcal{G})}\succ x\succ\overrightarrow{E(\mathcal{G})}\succ\overrightarrow{D'}\succ c\succ\overrightarrow{D''}$

(ii)-2: Reverse (ii)-1, and then shift c to the right by $B-1$ places and shift $V(\mathcal{G})\cup\{x\}\cup E(\mathcal{G})$ to the left by 1 place.

Let L be the score of c prior to executing any shift actions. Simple calculations show that each candidate in $V(\mathcal{G})\cup\{x\}\cup E(\mathcal{G})$ has score $L+B+1$, and each candidate in $F(\mathcal{G})\cup D'\cup D''$ has score at most $L+B$. Now, it is easy to show that Shift Bribery Is Safe for Borda is a no instance if and only if Multicoloured Independent Set is a yes instance.

Next, we obtain a similar result for Copeland, using a reduction from the Multicoloured Independent Set problem.

Theorem 4.4. *For Copeland, Shift Bribery Is Safe is* co-W[1]*-hard, when parameterized by the number of bribed voters.*

5 CONCLUSION

In this paper, we propose and study a nuanced notion of bribery which we call safe bribery. We observe that the computational complexity of safe bribery, for both \$Bribery and Shift Bribery, matches with the classical bribery problem for common voting rules. Hence, safety during bribery can often be achieved without incurring much additional computational overhead. However, we obtained some interesting results for k-approval, k-veto and simplified Bucklin, which were in P for the Shift Bribery problems but hard for \$Bribery problems. Our work is a natural extension of the bribery problem and can be further studied with respect to approximation algorithms and multi-winner voting rules.

REFERENCES

[1] Dorothea Baumeister, Piotr Faliszewski, Jérôme Lang, and Jörg Rothe. 2012. Campaigns for lazy voters: truncated ballots. In *Proc. 11th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*. 577–584.

[2] Dorothea Baumeister, Magnus Roos, and Jörg Rothe. 2011. Computational complexity of two variants of the possible winner problem. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. 853–860.

[3] Daniel Binkele-Raible, Gábor Erdélyi, Henning Fernau, Judy Goldsmith, Nicholas Mattei, and Jörg Rothe. 2014. The complexity of probabilistic lobbying. In *Algorithmic Decision Theory*, Vol. 11. Discrete Optimization, 1–21.

[4] Robert Bredereck, Jiehua Chen, Piotr Faliszewski, André Nichterlein, and Rolf Niedermeier. 2014. Prices Matter for the Parameterized Complexity of Shift Bribery. In *Proc. 28th AAAI Conference on Artificial Intelligence (AAAI)*. 1398–1404.

[5] Robert Bredereck, Piotr Faliszewski, Rolf Niedermeier, and Nimrod Talmon. 2016. Complexity of Shift Bribery in Committee Elections.. In *Proc. 30th AAAI Conference on Artificial Intelligence (AAAI)*. 2452–2458.

[6] Eric Brelsford, Piotr Faliszewski, Edith Hemaspaandra, Henning Schnoor, and Ilka Schnoor. 2008. Approximability of Manipulating Elections.. In *AAAI*, Vol. 8. 44–49.

[7] Lin Chen, Lei Xu, Shouhuai Xu, Zhimin Gao, Nolan Shah, Yang Lu, and Weidong Shi. 2018. Protecting Election from Bribery: New Approach and Computational Complexity Characterization. In *Proc. 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 1894–1896.

[8] Vincent Conitzer and Toby Walsh. 2016. Barriers to Manipulation in Voting. In *Handbook of Computational Social Choice*. 127–145. https://doi.org/10.1017/CBO9781107446984.007

[9] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2015. *Parameterized algorithms*. Vol. 5. Springer.

[10] Palash Dey. 2021. Local distance constrained bribery in voting. *Theor. Comput. Sci.* 849 (2021), 1–21. https://doi.org/10.1016/j.tcs.2020.10.005

[11] Palash Dey, Neeldhara Misra, and Y. Narahari. 2017. Frugal bribery in voting. *Theor. Comput. Sci.* 676 (2017), 15–32.

[12] Britta Dorn and Dominikus Krüger. 2016. On the hardness of bribery variants in voting with CP-nets. *Ann. Math. Artif. Intell.* 77, 3-4 (2016), 251–279. https://doi.org/10.1007/s10472-015-9469-3

[13] Britta Dorn and Ildikó Schlotter. 2012. Multivariate complexity analysis of swap bribery. *Algorithmica* 64, 1 (2012), 126–151.

[14] Jack Edmonds and Richard M Karp. 1972. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)* 19, 2 (1972), 248–264.

[15] Edith Elkind and Piotr Faliszewski. 2010. Approximation algorithms for campaign management. In *International Workshop on Internet and Network Economics*. Springer, 473–482.

[16] Edith Elkind, Piotr Faliszewski, and Arkadii Slinko. 2009. Swap bribery. In *Proc. 2nd International Symposium on Algorithmic Game Theory (SAGT 2009)*. Springer, 299–310.

[17] Edith Elkind, Piotr Faliszewski, and Arkadii Slinko. 2009. Swap Bribery. In *Algorithmic Game Theory*, Marios Mavronicolas and Vicky G. Papadopoulou (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 299–310.

[18] Gabor Erdelyi, Edith Hemaspaandra, and Lane A Hemaspaandra. 2014. Bribery and voter control under voting-rule uncertainty. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. 61–68.

[19] Piotr Faliszewski. 2008. Nonuniform Bribery. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3* (Estoril, Portugal) *(AAMAS '08)*. International Foundation for Autonomous Agents and Multiagent Systems, 1569–1572.

[20] Piotr Faliszewski, Rica Gonen, Martin Koutecký, and Nimrod Talmon. 2018. Opinion Diffusion and Campaigning on Society Graphs.. In *Proc. 27th International Joint Conference on Artificial Intelligence, IJCAI*. 219–225.

[21] Piotr Faliszewski, Edith Hemaspaandra, and Lane A Hemaspaandra. 2006. The complexity of bribery in elections. In *AAAI*, Vol. 6. 641–646.

[22] Piotr Faliszewski, Edith Hemaspaandra, and Lane A Hemaspaandra. 2009. How hard is bribery in elections? *Journal of artificial intelligence research* 35 (2009), 485–532.

[23] Piotr Faliszewski, Edith Hemaspaandra, Lane A Hemaspaandra, and Jörg Rothe. 2009. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research* 35 (2009), 275–341.

[24] Piotr Faliszewski, Edith Hemaspaandra, and Henning Schnoor. 2008. Copeland voting: Ties matter. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*. Citeseer, 983–990.

[25] Piotr Faliszewski, Yannick Reisch, Jörg Rothe, and Lena Schend. 2014. Complexity of manipulation, bribery, and campaign management in Bucklin and fallback voting. In *Proc. 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 1357–1358.

[26] Piotr Faliszewski, Yannick Reisch, Jörg Rothe, and Lena Schend. 2015. Complexity of manipulation, bribery, and campaign management in Bucklin and fallback voting. *Autonomous Agents and Multi-Agent Systems* 29, 6 (2015), 1091–1124.

[27] Piotr Faliszewski and Jörg Rothe. 2016. Control and Bribery in Voting. In *Handbook of Computational Social Choice*, Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia (Eds.). Cambridge University Press, 146–168. https://doi.org/10.1017/CBO9781107446984.008

[28] Delbert R Fulkerson. 1961. An out-of-kilter method for minimal-cost flow problems. *J. Soc. Indust. Appl. Math.* 9, 1 (1961), 18–27.

[29] Michael R. Garey and David S. Johnson. 1990. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA.

[30] Noam Hazon and Edith Elkind. 2010. Complexity of safe strategic voting. In *International Symposium on Algorithmic Game Theory*. Springer, 210–221.

[31] Egor Ianovski, Lan Yu, Edith Elkind, and Mark C Wilson. 2011. The complexity of safe manipulation under scoring rules. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

[32] Benjamin N Jackson, Patrick S Schnable, and Srinivas Aluru. 2008. Consensus genetic maps as median orders from inconsistent sources. *IEEE/ACM Transactions on computational biology and bioinformatics* 5, 2 (2008), 161–171.

[33] Andrzej Kaczmarczyk and Piotr Faliszewski. 2016. Algorithms for destructive shift bribery. In *Proc. 15th International Conference on Autonomous Agents & Multiagent Systems (AAMAS)*. 305–313.

[34] Neel Karia, Faraaz Mallick, and Palash Dey. 2022. How Hard is Safe Bribery? *arXiv preprint arXiv:2201.10383* (2022).

[35] Orgad Keller, Avinatan Hassidim, and Noam Hazon. 2018. Approximating Bribery in Scoring Rules. In *Proc. 32nd International Conference on Artificial Intelligence (AAAI)*. 1121–1129.

[36] Dušan Knop, Martin Koutecký, and Matthias Mnich. 2018. A Unifying Framework for Manipulation Problems. In *Proc. 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 256–264.

[37] Luke Mathieson and Stefan Szeider. 2012. Editing graphs to satisfy degree constraints: A parameterized approach. *J. Comput. System Sci.* 78, 1 (2012), 179–191. https://doi.org/10.1016/j.jcss.2011.02.001 JCSS Knowledge Representation and Reasoning.

[38] Nicholas Mattei, Maria Silvia Pini, Francesca Rossi, and Kristen Brent Venable. 2012. Bribery in Voting Over Combinatorial Domains Is Easy.. In *ISAIM*.

[39] Cynthia Maushagen, Marc Neveling, Jörg Rothe, and Ann-Kathrin Selker. 2018. Complexity of Shift Bribery in Iterative Elections. In *Proc. 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. 1567–1575.

[40] Svetlana Obraztsova and Edith Elkind. 2012. Optimal Manipulation of Voting Rules. In *Proc. 26th AAAI Conference on Artificial Intelligence (AAAI)*. 2141–2147.

[41] Svetlana Obraztsova and Edith Elkind. 2012. Optimal manipulation of voting rules. In *Proc. 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. 619–626.

[42] David M. Pennock, Eric Horvitz, and C. Lee Giles. 2000. Social Choice Theory and Recommender Systems: Analysis of the Axiomatic Foundations of Collaborative Filtering. In *Proc. 17th National Conference on Artificial Intelligence and 12th Conference on on Innovative Applications of Artificial Intelligence, July 30 - August 3, 2000, Austin, Texas, USA*. 729–734. http://www.aaai.org/Library/AAAI/2000/aaai00-112.php

[43] Maria Silvia Pini, Francesca Rossi, and Kristen Brent Venable. 2013. Bribery in voting with soft constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 27.

[44] Ildikó Schlotter, Piotr Faliszewski, and Edith Elkind. 2011. Campaign management under approval-driven voting rules. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 25.

[45] Ildikó Schlotter, Piotr Faliszewski, and Edith Elkind. 2017. Campaign management under approval-driven voting rules. *Algorithmica* 77, 1 (2017), 84–115.

[46] Arkadii Slinko and Shaun White. 2008. Nondictatorial social choice rules are safely manipulable. In *Proceedings of the Second International Workshop on Computational Social Choice (COMSOC-2008)*. 403–414.

[47] Arkadii Slinko and Shaun White. 2014. Is it ever safe to vote strategically? *Social Choice and Welfare* 43, 2 (2014), 403–427.

[48] Lirong Xia. 2012. Computing the margin of victory for various voting rules. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. 982–999.