Multivariate Complexity Analysis of Swap Bribery

Britta Dorn[1] and Ildikó Schlotter[2⋆]

[1] Wilhelm-Schickard-Institut für Informatik, Universität Tübingen
Sand 13, 72076 Tübingen, Germany
bdorn@informatik.uni-tuebingen.de
[2] Budapest University of Technology and Economics
H-1521 Budapest, Hungary
ildi@cs.bme.hu

Abstract. We consider the computational complexity of a problem modeling *bribery* in the context of voting systems. In the scenario of Swap Bribery, each voter assigns a certain price for swapping the positions of two consecutive candidates in his preference ranking. The question is whether it is possible, without exceeding a given budget, to bribe the voters in a way that the preferred candidate wins in the election. We initiate a parameterized and multivariate complexity analysis of Swap Bribery, focusing on the case of k-approval. We investigate how different cost functions affect the computational complexity of the problem. We identify a special case of k-approval for which the problem can be solved in polynomial time, whereas we prove NP-hardness for a slightly more general scenario. We obtain fixed-parameter tractability as well as W[1]-hardness results for certain natural parameters.

1 Introduction

In the context of voting systems, the question of how to manipulate the votes in some way in order to make a preferred candidate win the election is a very interesting question. One possibility is *bribery*, which can be described as spending money on changing the voters' preferences over the candidates in such a way that a preferred candidate wins, while respecting a given budget. There are various situations that fit into this scenario: The act of bribing the voters in order to make them change their preferences, or paying money in order to get into the position of being able to change the submitted votes, but also the setting of systematically spending money in an election campaign in order to convince the voters to change their opinion on the ranking of candidates.

The study of bribery in the context of voting systems was initiated by Faliszewski, Hemaspaandra, and Hemaspaandra in 2006 [11]. Since then, various models have been analyzed. In the original version, each voter may have a different but fixed price which is independent of the changes made to the bribed vote. The scenario of nonuniform bribery introduced by Faliszewski [10] and the case of microbribery studied by Faliszewski, Hemaspaandra, Hemaspaandra, and Rothe in [12] allow for prices that depend on the amount of change the voter is asked for by the briber.

In addition, the Swap Bribery problem as introduced by Elkind, Faliszewski, and Slinko [9] takes into consideration the ranking aspect of the votes: In this model, each voter may assign different prices for swapping two consecutive candidates in his preference ordering. This approach is natural, since it captures the notion of small changes and comprises the preferences of the voters. Elkind et al. [9] prove complexity results for this problem for several election systems such as Borda, Copeland, Maximin, and approval voting. In particular, they provide a detailed case study for k-approval. In this voting system, every voter

⋆ Supported by the Hungarian National Research Fund (OTKA 67651).

	Result	Reference
$k = 1$ or $k = m - 1$	P	[9]
$1 \leq k \leq m$, m or n constant	P	[9]
$1 \leq k \leq m$, all costs $= 1$	**P**	Thm. 1
$k = 2$	NP-complete	[2]
$3 \leq k \leq m - 2$, costs in $\{0, 1, 2\}$	NP-complete	[9]
$2 \leq k \leq m - 2$, costs in $\{0, 1\}$ and $\beta = 0$	**NP-complete**	[2], Prop. 2
$2 \leq k \leq m - 2$ is part of the input, costs in $\{0, 1\}$ and $\beta = 0$, n constant	**NP-complete**	[3], Prop. 2
$2 \leq k \leq m - 2$, costs in $\{1, 1 + \varepsilon\}$, $\varepsilon > 0$	**NP-complete**, **W[1]-hard** (β)	Thm. 3
$1 \leq k \leq m$	**FPT** (m)	Thm. 4
$1 \leq k \leq m$ is part of the input	**FPT** (β, n) **by kernelization**	Thm. 5
$1 \leq k \leq m$	**FPT** (β, n, k) **by kernelization**	Thm. 5

Table 1. Overview of known and new results for Swap Bribery for k-approval. The results obtained in this paper are printed in bold. Here, m and n denote the number of candidates and votes, respectively, and β is the budget. For the parameterized complexity results, the parameters are indicated in brackets. If not stated otherwise, the value of k is fixed.

can specify a group of k preferred candidates which are assigned one point each, whereas the remaining candidates obtain no points. The candidates which obtain the highest sum of points over all votes are the winners of the election. Two prominent special cases of k-approval are plurality, (where $k = 1$, i.e., every voter can vote for exactly one candidate) and veto (where $k = m - 1$ for m candidates, i.e., every voter assigns one point to all but one disliked candidate). Table 1 shows a summary of research considering Swap Bribery for k-approval, including both previously known and newly achieved results.

This paper contributes to the further investigation of the case study of k-approval that was initiated in [9], this time from a parameterized point of view. This approach seems to be appealing in the context of voting systems, where NP-hardness is a desired property for various problems, like Manipulation (where certain voters, the manipulators, know the preferences of the remaining voters and try to adjust their own preferences in such a way that a preferred candidate wins), Lobbying (here, a lobby affects certain voters on their decision for several issues in an election), Control (where the chair of the election tries to make a certain candidate win (or lose) by deleting or adding either candidates or votes), or, as in our case, Swap Bribery. However, NP-hardness does not necessarily constitute a guarantee against such dishonest behavior. As Conitzer et al. [7] point out for the Manipulation problem, an NP-hardness result in these settings would lose relevance if an efficient fixed-parameter algorithm with respect to an appropriate parameter was found. Parameterized complexity can hence provide a more robust notion of hardness. The investigation of problems from voting theory under this aspect has started, see for example [1, 3, 4, 6, 18].

We show NP-hardness as well as fixed-parameter intractability of Swap Bribery for certain very restricted cases of k-approval if the parameter is the budget, whereas we identify a natural special case of the problem which can be solved in polynomial time. By contrast, we obtain fixed-parameter tractability with respect to the parameter 'number of candidates' for k-approval and a large class of other voting systems, and a polynomial kernel for k-approval if we consider certain combined parameters.

The paper is organized as follows. After introducing notation in Section 2, we investigate the complexity of Swap Bribery depending on the cost function in Section 3, where we show the connection to the Possible Winner problem, identify a polynomial-time solvable case of k-approval and a hardness result. In Section 4, we consider the parameter 'number of candidates' and obtain an FPT result for Swap Bribery for a large class of voting systems. We also consider the combination of parameters 'number of votes' and 'size of the budget'. We conclude with a discussion of open problems and further directions that might be interesting for future investigations.

2 Preliminaries

Elections. An $\mathcal{E}$-*election* is a pair $E = (C, V)$, where $C = \{c_1, \dots, c_m\}$ denotes the set of *candidates*, $V = \{v_1, \dots, v_n\}$ is the set of *votes* or *voters*, and $\mathcal{E}$ is the *election system* which is a function mapping (C, V) to a set $W \subseteq C$ called the *winners* of the election. We will express our results for the *winner case* where several winners are possible, but our results can be adapted to the *unique winner case* where W consists of a single candidate only.

In our context, each vote is a strict linear order over the set C, and we denote by $\text{rank}(c, v)$ the position of candidate $c \in C$ in a vote $v \in V$.

For an overview of different election systems, we refer to [5]. We will mainly focus on election systems that are characterized by a given *scoring rule*, expressed as a vector $(s_1, s_2, \dots, s_m)$. Given such a scoring rule, the *score* of a candidate c in a vote v, denoted by $\text{score}(c, v)$, is $s_{\text{rank}(c,v)}$. The score of a candidate c in a set of votes V is $\text{score}(c, V) = \sum_{v \in V} \text{score}(c, v)$, and the winners of the election are the candidates that receive the highest score in the given votes.

The election system we are particularly interested in is k-approval, which is defined by the scoring vector $(1, \dots, 1, 0, \dots, 0)$, starting with k ones. In the case of $k = 1$, this is the *plurality* rule, whereas $(m-1)$-approval is also known as *veto*. Given a vote v, we will say that a candidate c with $1 \leq \text{rank}(c, v) \leq k$ takes a *one-position* in v, whereas a candidate c' with $k+1 \leq \text{rank}(c', v) \leq m$ takes a *zero-position* in v.

Swap Bribery, Possible Winner, Manipulation. Given V and C, a *swap* in some vote $v \in V$ is a triple (v, c_1, c_2) where $\{c_1, c_2\} \subseteq C, c_1 \neq c_2$. Given a vote v, we say that a swap $\gamma = (v, c_1, c_2)$ is *admissible in* v, if $\text{rank}(c_1, v) = \text{rank}(c_2, v) - 1$. Applying this swap means exchanging the positions of c_1 and c_2 in the vote v, we denote by v^γ the vote obtained this way. Given a vote v, a set Γ of swaps is *admissible in* v, if the swaps in Γ can be applied in v in a sequential manner, one after the other, in some order. Note that the obtained vote, denoted by v^Γ, is independent from the order in which the swaps of Γ are applied. We also extend this notation for applying swaps in several votes, in the straightforward way.

In a Swap Bribery instance, we are given V, C, and $\mathcal{E}$ forming an election, a preferred candidate $p \in C$, a cost function $c : C \times C \times V \to \mathbb{N}$ mapping each possible swap to a non-negative integer, and a budget $\beta \in \mathbb{N}$. The task is to determine a set of admissible swaps Γ whose total cost is at most β, such that p is a winner in the $\mathcal{E}$-election (C, V^Γ). Such a set of swaps is called a *solution* of the Swap Bribery instance. The underlying decision problem is the following.

Swap Bribery
Given: An $\mathcal{E}$-election $E = (C, V)$, a preferred candidate $p \in C$, a cost function c mapping each possible swap to a non-negative integer, and a budget $\beta \in \mathbb{N}$.
Question: Is there a set of swaps Γ whose total cost is at most β such that p is a winner in the $\mathcal{E}$-election (C, V^Γ)?

We will also show the connection between Swap Bribery and the Possible Winner problem. In this setting, we have an election where some of the votes may be *partial* orders over C instead of complete linear ones. The question is whether it is possible to extend the partial votes to complete linear orders in such a way that a preferred candidate wins the election. For a more formal definition, we refer to the article by Konczak and Lang [16] who introduced this problem. The corresponding decision problem is defined as follows.

Possible Winner
Given: A set of candidates C, a set of partial votes $V' = (v'_1, \dots, v'_n)$ over C, an election system $\mathcal{E}$, and a preferred candidate $p \in C$.
Question: Is there an extension $V = (v_1, \dots, v_n)$ of V' such that each v_i extends v'_i to a complete linear order, and p is a winner in the $\mathcal{E}$-election (C, V)?

A special case of Possible Winner is Manipulation (see e.g. [7, 15]). Here, the given set of partial orders consists of two subsets; one subset contains complete preference orders and the other one completely unspecified votes.

Parameterized complexity, Multivariate complexity. We assume the reader to be familiar with the concepts of parameterized complexity, parameterized reductions and kernelization [8, 14, 19]. Multivariate complexity is the natural sequel of the parameterized approach when expanding to multidimensional parameter spaces, see the recent survey by Niedermeier [20]. For the hardness reduction in Theorem 3, we will use the following W[1]-hard problem [13]:

Multicolored Clique
Given: An undirected graph $G = (V_1 \cup V_2 \cup \dots \cup V_k, E)$ with $V_i \cap V_j = \emptyset$ for $1 \leq i < j \leq k$ where the vertices of V_i induce an independent set for $1 \leq i \leq k$.
Question: Is there a complete subgraph (clique) of G of size k?

3 Complexity depending on the cost function

In this section, we focus our attention on Swap Bribery for k-approval. We start with the case where all costs are equal to 1, for which we obtain polynomial-time solvability.

Theorem 1. Swap Bribery *for k-approval is polynomial-time solvable, if all costs are* 1.

Theorem 1 provides an algorithm which checks for every possible s, if there is a solution in which the preferred candidate wins with score s. This can be carried out by solving a minimum cost maximum flow problem. Due to lack of space, we omit the proof of Theorem 1; see the Appendix for it.

Note that Theorem 1 also implies a polynomial-time approximation algorithm for Swap Bribery for k-approval with approximation ratio δ, if all costs are in $\{1, \delta\}$ for some $\delta \geq 1$.

Proposition 2 shows the connection between Swap Bribery and Possible Winner. This result is an easy consequence of a reduction given by Elkind et al. [9]. For the proof of the other direction, see again the Appendix.

Proposition 2. *The special case of* Swap Bribery *where the costs are in $\{0, \delta\}$ for some $\delta > 0$ and the budget is zero is equivalent to the* Possible Winner *problem.*

As a corollary, Swap Bribery with costs in $\{0, \delta\}$, $\delta > 0$ and budget zero is NP-complete for almost all election systems based on scoring rules [2]. For many voting systems such as k-approval, Borda, and Bucklin, it is NP-complete even for a fixed number of votes [3].

We now turn to the case with two different positive costs, addressing 2-approval.

Theorem 3. *Suppose that $\varepsilon > 0$.*
(1) Swap Bribery *for 2-approval with costs in $\{1, 1 + \varepsilon\}$ is NP-complete.*
(2) Swap Bribery *for 2-approval with costs in $\{1, 1 + \varepsilon\}$ is W[1]-hard, if the parameter is the budget β, or equivalently, the maximum number of swaps allowed.*

Proof. We present a reduction from the MULTICOLORED CLIQUE problem. Let $\mathcal{G} = (V, E)$ with the k-partition $V = V_1 \cup V_2 \cup \dots \cup V_k$ be the given instance of MULTICOLORED CLIQUE. Here and later, we write $[k]$ for $\{1, 2, \dots, k\}$. For each $i \in [k]$, $x \in V_i$, and $j \in [k] \setminus \{i\}$ we let $E_x^j = \{xy \mid y \in V_j, xy \in E\}$. We construct an instance $I_\mathcal{G}$ of SWAP BRIBERY as follows.

The set $\mathcal{C}$ of candidates will be the union of the sets $A, B, C, \widetilde{C}, F, H, \widetilde{H}, M, \widetilde{M}, D, G, T$, and $\{p, r\}$ where

$A = \{a^{i,j} \mid i, j \in [k]\}$,
$B = \{b_v^j \mid j \in [k], v \in V\}$, and $C, \widetilde{C}, F, H$ are defined analogously,
$\widetilde{H} = \{h_v^j \mid j \in [k], v \in \bigcup_{i<j} V_i\}$,
$M = \{m^{i,j} \mid 1 \le i \le j \le k\}$,
$\widetilde{M} = \{\widetilde{m}^{i,j} \mid 1 \le i < j \le k\}$.

Our preferred candidate is p. The sets $D = \{d_1, d_2, \dots\}$, $G = \{g_1, g_2, \dots\}$, and $T = \{t_1, t_2, \dots\}$ will contain *dummies*, *guards*, and *transporters*, respectively. Our budget will be $\beta = k^3 + 10k^2$. Regarding the indices i and j, we suppose $i, j \in [k]$ if not stated otherwise.

The set of votes will be $W = W_G \cup W_I \cup W_S \cup W_C$. Votes in W_G will define guards (explained later), votes in W_I will set the initial scores, votes in W_S will represent the selection of k vertices, and finally, votes in W_C will be responsible for checking that the selected vertices are pairwise neighboring. We construct W such that the following will hold for some (even) integer K, determined later:

$\text{score}(r, W) = 0$,
$\text{score}(a, W) = K + 1$ for each $a \in A$,
$\text{score}(q, W) = K$ for each $q \in \mathcal{C} \setminus (A \cup D \cup \{r\})$,
$\text{score}(d, W) \le 1$ for each $d \in D$.

We define the cost function c such that each swap has cost 1 or $1 + \varepsilon$. We will define each cost to be 1 if not explicitly stated otherwise. Since each cost is at least 1, none of the candidates ranked after the position $\beta + 2$ in a vote v can receive non-zero points in v without violating the budget. Thus, we can represent votes by listing only their first $\beta + 2$ positions. A candidate does not *appear* in some vote, if he is not contained in these positions.

Dummies, guards, truncation, and transporters. First, let us clarify the concept of dummy candidates: we will ensure that no dummy can receive more than one point in total, by letting each $d \in D$ appear in exactly one vote. Since we will use at most one dummy in each vote, this can be ensured easily by using at most $|W|$ dummies in total. We will use the sign $*$ to denote dummies in votes.

Now, we define $\beta + 2$ guards using the votes W_G. We let W_G contain votes of the form $w_G(h)$ for each $h \in [\beta + 2]$, each such vote having multiplicity $K/2$ in W_G. We let $w_G(h) = (g_h, g_{h+1}, g_{h+2}, \dots, g_{\beta+2}, g_1, g_2, \dots g_{h-1})$. Note that $\text{score}(g, W_G) = K$ for each $g \in G$, and the total score obtained by the guards in W_G cannot decrease. As we will make sure that p cannot receive more than K points without exceeding the budget, this yields that in any possible solution, each guard must have score exactly K.

Using guards, we can *truncate* votes at any position $h > 2$ by putting arbitrarily chosen guards at the positions $h, h + 1, \dots, \beta + 2$. This way we ensure that only candidates on the first $h - 1$ positions can receive a point in this vote. We will denote truncation at position h by using a sign † at that position.

Sometimes we will need votes which ensure that some candidate q_1 can "transfer" one point to some candidate q_2 using a cost of c from the budget ($c \in \mathbb{N}^+$, $q_1, q_2 \in \mathcal{C} \setminus (D \cup D \cup G)$). In such cases, we construct c votes by using exactly $c - 1$ *transporter* candidates, say $t_1, t_2, \dots, t_{c-1}$, none of which appears in any other vote in $W \setminus W_I$. The constructed votes are as follows: for each $h \in [c - 2]$ we add a vote $(*, t_h, t_{h+1}, \dagger)$, and we also add the votes $(*, q_1, t_1, \dagger)$ and $(*, t_{c-1}, q_2, \dagger)$. We let the cost of any swap here be 1, and we denote the obtained set of votes by $q_1 \rightsquigarrow^c q_2$. (Note that $q_1 \rightsquigarrow^1 q_2$ only consists of the vote $(*, q_1, q_2, \dagger)$.)

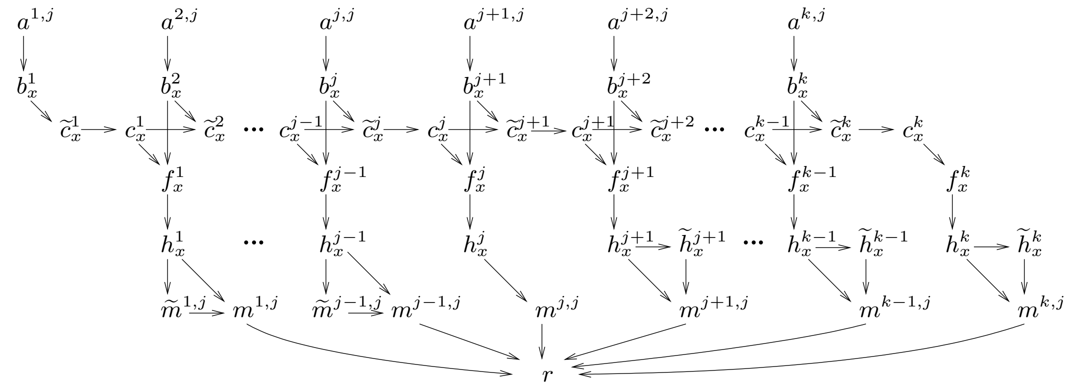

Fig. 1. Part of the instance $I_\mathcal{G}$ in the proof of Theorem 3, assuming $x \in V_j$ in the figure. An arc goes from q_1 to q_2 if q_1 can transfer a point to q_2 using one or several swaps.

Observe that the votes $q_1 \rightsquigarrow^c q_2$ ensure that q_1 can transfer one point to q_2 at cost c. Later, we will make sure $\text{score}(t, W) = K$ for each transporter $t \in T$. Thus, no transporter can increase its score in a solution, and q_1 only loses a point in these votes if q_2 gets one.

Setting initial scores. Using dummies and guards, we define W_I to adjust the initial scores of the relevant candidates as follows. We put the following votes into W_I:

$(p, *, \dagger)$ with multiplicity K,	
$(a^{i,j}, *, \dagger)$ with multiplicity $K+1-\lvert V_j\rvert$	for each $i, j \in [k]$,
$(h_x^i, *, \dagger)$ with multiplicity $K - \lvert E_x^i\rvert$	for each $i \in [k], x \in \bigcup_{i<j} V_j$,
$(\widetilde{h}_x^i, *, \dagger)$ with multiplicity $K - \lvert E_x^i\rvert$	for each $i \in [k], x \in \bigcup_{i>j} V_j$,
$(m^{i,j}, *, \dagger)$ with multiplicity $K-2$	for each $i < j$,
$(q, *, \dagger)$ with multiplicity $K-1$	for each remaining $q \notin D \cup G \cup \{r\}$.

The preferred candidate p will not appear in any other vote, implying $\text{score}(p, W) = K$.

Selecting vertices. The set W_S consists of the following votes:

$a^{i,j} \rightsquigarrow^1 b_x^i$	for each $i, j \in [k]$ and $x \in V_j$,
$b_x^1 \rightsquigarrow^2 \widetilde{c}_x^1$	for each $x \in V$,
$w_S(i, x) = (b_x^i, c_x^{i-1}, \widetilde{c}_x^i, f_x^{i-1}, \dagger)$	for each $2 \le i \le k$, $x \in V$,
$c_x^k \rightsquigarrow^2 f_x^k$	for each $x \in V$,
$\widetilde{c}_x^i \rightsquigarrow^1 c_x^i$	for each $i \in [k], x \in V$, and
$f_x^i \rightsquigarrow^{2(k-i)+1} h_x^i$	for each $i \in [k], x \in V$.

Swapping candidate b_x^i with c_x^{i-1}, and swapping candidate $\widetilde{c}_x^i$ with f_x^{i-1} in $w_S(i, x)$ for some $2 \le i \le k$, $x \in V$ will have cost $1 + \varepsilon$.

Checking incidency. The set W_C will contain the votes

$h_x^i \rightsquigarrow^1 \widetilde{h}_x^i$	for each $i \in [k]$, $x \in \bigcup_{i>j} V_j$,
$w_C(i, j, y, x) = (h_x^i, \widetilde{h}_y^j, \widetilde{m}^{i,j}, m^{i,j}, \dagger)$	for each $i < j$, $x \in V_j$, $y \in V_i$, $xy \in E$,
$\widetilde{m}^{i,j} \rightsquigarrow^1 m^{i,j}$	for each $i < j$,
$h_x^i \rightsquigarrow^3 m^{i,i}$	for each $i \in [k]$, $x \in V_i$, and
$m^{i,j} \rightsquigarrow^1 r$ with multiplicity 2	for each $i < j$,
$m^{i,i} \rightsquigarrow^1 r$	for each $i \in [k]$.

Again, swapping candidate h_x^i with $\widetilde{h}_y^j$, and also candidate $\widetilde{m}^{i,j}$ with $m^{i,j}$ in a vote of the form $w_C(i, j, y, x)$ will have cost $1 + \varepsilon$.

It remains to define K properly. To this end, we let $K \ge 2$ be the minimum even integer not smaller than the integers in the set $\{\lvert E_x^j\rvert \mid j \in [k], x \notin V_j\} \cup \{\lvert V_i\rvert \mid i \in [k]\} \cup \{k^2\}$.

This finishes the construction. It is straightforward to verify that the initial scores of the candidates are as claimed above. The constructed instance is illustrated in Fig. 1.

Construction time. Note $|W_G| = (\beta+2)K/2$, $|W_I| = O(Kk^2 + Kk|V|) = O(Kk|V|)$, $|W_S| = O(k^2|V|)$, and $|W_C| = O(k|V|+|E|)$. Hence, the number of votes is polynomial in the size of the input graph $\mathcal{G}$. This also implies that the number of candidates is polynomial as well, and the whole construction takes polynomial time. Note also that β is only a function of k, hence this yields an FPT reduction as well.

If for some vote v, exactly one candidate q_1 gains a point and exactly one candidate q_2 loses a point as a result of the swaps in Γ, then we say that q_2 *sends* one point to q_1, or equivalently, q_1 *receives* one point from q_2 in v according to Γ. Also, if Γ consists of swaps that transform a vote $(a, b, c, d, \dagger)$ into a vote $(c, d, a, b, \dagger)$, then we say that a sends one point to c, and b sends one point to d. A point is *transferred* from q_1 to q_2 in Γ, if it is sent from q_1 to q_2 possibly through some other candidates.

Our aim is to show the following: $\mathcal{G}$ has a k-clique if and only if the constructed instance $I_\mathcal{G}$ is a yes-instance of Swap Bribery. This will prove both (1) and (2).

Direction $\Longrightarrow$. Suppose that $\mathcal{G}$ has a clique consisting of the vertices $x_1, x_2, \dots, x_k$ with $x_i \in V_i$. We are going to define a set Γ of swaps transforming W into W^Γ with total cost β such that p wins in W^Γ according to 2-approval.

First, we define the swaps applied by Γ in W_S:

- Swap $a^{i,j}$ with $b^i_{x_j}$ for each i and j in $a^{i,j} \rightsquigarrow^1 b^i_{x_j}$. Cost: k^2.
- Transfer one point from $b^1_{x_j}$ to $\widetilde{c}^1_{x_j}$ for each $j \in [k]$ in $b^1_{x_j} \rightsquigarrow^2 \widetilde{c}^1_{x_j}$. Cost: $2k$.
- Apply four swaps in each vote $w_S(i, x_j) = (b^i_{x_j}, c^{i-1}_{x_j}, \widetilde{c}^i_{x_j}, f^{i-1}_{x_j}, \dagger)$ transforming it to $(\widetilde{c}^i_{x_j}, f^{i-1}_{x_j}, b^i_{x_j}, c^{i-1}_{x_j}, \dagger)$, sending one point from $b^i_{x_j}$ to $\widetilde{c}^i_{x_j}$ and simultaneously, also one point from $c^{i-1}_{x_j}$ to $f^{i-1}_{x_j}$. Cost: $4k(k-1)$.
- Swap $\widetilde{c}^i_{x_j}$ with $c^i_{x_j}$ for each $i, j \in [k]$ in $\widetilde{c}^i_{x_j} \rightsquigarrow^1 c^i_{x_j}$. Cost: k^2.
- Transfer one point from $c^k_{x_j}$ to $f^k_{x_j}$ for each $j \in [k]$ in $c^k_{x_j} \rightsquigarrow^2 f^k_{x_j}$. Cost: $2k$.
- Transfer one point from $f^i_{x_j}$ to $h^i_{x_j}$ for each $i, j \in [k]$ in $f^i_{x_j} \rightsquigarrow^{2(k-i)+1} h^i_{x_j}$. Cost: k^3.

The above swaps transfer one point from $a^{i,j}$ to $h^i_{x_j}$ via the candidates $b^i_{x_j}$, $\widetilde{c}^i_{x_j}$, $c^i_{x_j}$, and $f^i_{x_j}$ for each i and j. These swaps of Γ, applied in the votes W_S, have total cost $k^3 + 6k^2$.

Now, we define the swaps applied by Γ in the votes W_C.

- Swap $h^i_{x_j}$ with $\widetilde{h}^i_{x_j}$ for each $j < i$ in $h^i_{x_j} \rightsquigarrow^1 \widetilde{h}^i_{x_j}$. Cost: $k(k-1)/2$.
- Apply four swaps in each vote $w_C(i, j, x_i, x_j) = (h^i_{x_j}, \widetilde{h}^j_{x_i}, \widetilde{m}^{i,j}, m^{i,j}, \dagger)$ transforming it to $(\widetilde{m}^{i,j}, m^{i,j}, h^i_{x_j}, \widetilde{h}^j_{x_i}, \dagger)$, sending one point from $h^i_{x_j}$ to $\widetilde{m}^{i,j}$ and, simultaneously, also one point from $\widetilde{h}^j_{x_i}$ to $m^{i,j}$. Note that $w_C(i, j, x_i, x_j)$ is indeed defined for each i and j, since x_i and x_j are neighboring. Cost: $2k(k-1)$.
- Swap $\widetilde{m}^{i,j}$ with $m^{i,j}$ for each $i < j$ in $\widetilde{m}^{i,j} \rightsquigarrow^1 m^{i,j}$. Cost: $k(k-1)/2$.
- Transfer one point from $h^i_{x_i}$ to $m^{i,i}$ for each $i \in [k]$ in $h^i_{x_i} \rightsquigarrow^3 m^{i,i}$. Cost: $3k$.
- Swap $m^{i,j}$ with r in both of the votes $m^{i,j} \rightsquigarrow^1 r$ for each $i < j$. Cost: $k(k-1)$.
- Swap $m^{i,i}$ with r for each $i \in [k]$ in $m^{i,i} \rightsquigarrow^1 r$. Cost: k.

Candidate r receives k^2 points after all these swaps in Γ. Easy computations show that the above swaps have cost $4k^2$, so the total cost of Γ is $\beta = k^3 + 10k^2$. Clearly,

$\text{score}(p, W^\Gamma) = K$,
$\text{score}(r, W^\Gamma) = k^2 \le K$,
$\text{score}(a, W^\Gamma) = K$ for each $a \in A$, and
$\text{score}(q, W^\Gamma) = \text{score}(q, W) \le K$ for all the remaining candidates q.

This means that p is a winner in W^Γ according to 2-approval. Hence, Γ is indeed a solution for $I_\mathcal{G}$, proving the first direction of the reduction.

Direction $\Longleftarrow$. Suppose that $I_{\mathcal{G}}$ is solvable, and there is a set Γ of swaps transforming W into W^Γ with total cost at most β such that p wins in W^Γ according to 2-approval. We also assume w.l.o.g. that Γ is a solution having minimum cost.

As argued above, $\text{score}(p, W^\Gamma) \leq K$ and $\text{score}(g, W^\Gamma) \geq K$ for each $g \in G$ follow directly from the construction. Thus, only $\text{score}(p, W^\Gamma) = \text{score}(g, W^\Gamma) = K$ for each $g \in G$ is possible. Hence, for any $i, j \in [k]$, by $\text{score}(a^{i,j}, W) = K + 1$ we get that $a^{i,j}$ must lose at least one point during the swaps in Γ. As no dummy can have more points in W^Γ than in W (by their positions), and each candidate in $\mathcal{C} \setminus (A \cup D \cup \{r\})$ has K points in W, the k^2 points lost by the candidates in A can only be transferred by Γ to the candidate r.

By the optimality of Γ, this means that $a^{i,j}$ sends a point to b^i_x in Γ for some unique $x \in V_j$; we define $\sigma(i,j) = x$ in this case. First, we show $\sigma(1,j) = \sigma(2,j) = \cdots = \sigma(k,j)$ for each $j \in [k]$, and then we prove that the vertices $\sigma(1,1), \ldots, \sigma(k,k)$ form a k-clique in $\mathcal{G}$.

Let B^* be the set of candidates in B that receive a point from some candidate in A according to Γ; $|B^*| = k^2$ follows from the minimality of Γ. Observing the votes in $W_S \cup W_C$, we can see that some $b^i_x \in B^*$ can only transfer one point to r by transferring it to $h^{i'}_x$ via $f^{i'}_x$ for some i' using swaps in the votes W_S, and then transferring the point from $h^{i'}_x$ to r using swaps in the votes W_C. Basically, there are three ways to transfer a point from b^i_x to $h^{i'}_x$:

(A) b^i_x sends one point to f^{i-1}_x in $w_S(i,x)$ at a cost of $3 + 2\varepsilon$, and then f^{i-1}_x transfers one point to h^{i-1}_x. This can be carried out applying exactly $3 + 2(k-i+1) + 1 = 6 + 2(k-i)$ swaps, having total costs $6 + 2(k-i) + 2\varepsilon$.

(B) b^i_x sends one point to $\widetilde{c}^i_x$ in $w_S(i,x)$, $\widetilde{c}^i_x$ sends one point to c^i_x, c^i_x sends one point to f^i_x, and then the point gets transferred to h^i_x. Again, the number of used swaps is exactly $5 + 2(k-i) + 1 = 6 + 2(k-i)$, and the total cost is at least $6 + 2(k-i)$.

(C) b^i_x sends one point to $\widetilde{c}^i_x$ in $w_S(i,x)$, and then the point is transferred to a candidate $f^{i'}_x$ for some $i' > i$ via the candidates $c^i_x, \widetilde{c}^{i+1}_x, c^{i+1}_x, \ldots, c^{i'}_x$. Again, the number of used swaps is exactly $5 + 2(k-i) + 1 = 6 + 2(k-i)$, and the total cost is at least $6 + 2(k-i)$.

Summing up these costs for each $b^i_x \in B^*$, and taking into account the cost of sending the k^2 points from the candidates of A to B^*, we get that the swaps of Γ applied in the votes W_S must have total cost at least $k^2 + k\left(\sum_{j=1}^{k} 6 + 2(k-i)\right) = k^3 + 6k^2$. Equality can only hold if each $b^i_x \in B^*$ transfers one point to $h^{i'}_x$ for some $i' \geq i$, i.e. either case B or C happens.

Let H^* be the set of those k^2 candidates in H that receive a point transferred from a candidate in B^*, and let us consider now the swaps of Γ applied in the votes W_C that transfer one point from a candidate $h^i_x \in H^*$ to r. Let j be the index such that $x \in V_j$. First, note that h^i_x must transfer one point to $m^{i,j}$ (if $i \leq j$) or to $m^{j,i}$ (if $i > j$). Moreover, independently of whether $i < j$, $i = j$, or $i > j$ holds, this can only be done using exactly 3 swaps, thanks to the role of the candidates in $\widetilde{H}$ and in $\widetilde{M}$. To see this, note that only the below possibilities are possible:

- If $i < j$, then h^i_x sends one point in $w_C(i,j,y,x)$ for some $y \in V_i$ either to $\widetilde{m}^{i,j}$ via two swaps, or to $m^{i,j}$ via three swaps. In the former case, $\widetilde{m}^{i,j}$ must further transfer the point to $m^{i,j}$, which is the third swap needed.
- If $i > j$, then h^i_x first sends one point to $\widetilde{h}^i_x$, and then $\widetilde{h}^i_x$ sends this point either to $\widetilde{m}^{j,i}$ via one swap, or to $m^{j,i}$ via two swaps applied in the vote $w_C(j,i,x,y)$ for some $y \in V_i$. In the former case, $\widetilde{m}^{j,i}$ transfers the point to $m^{j,i}$ via an additional swap. Note that in any of these cases, Γ applies 3 swaps (maybe having cost $3 + \varepsilon$ or $3 + 2\varepsilon$).
- If $i = j$, then h^i_x sends one point to $m^{i,i}$ through 3 swaps.

Thus, transferring a point from h^i_x to r needs 4 swaps in total, and hence the number of swaps applied by Γ in the votes W_C is at least $4k^2$. Now, by $\beta = k^3 + 10k^2$ we know that equality must hold everywhere in the previous reasonings. Therefore, as argued above, each b_x^i must transfer a point to $h_x^{i'}$ for some $i' \geq i$, i.e., only cases B and C might happen from the above listed possibilities. Now, we are going to argue that only case B can occur.

Let us consider the multiset I_B containing k^2 pairs of indices, obtained by putting (i, j) into I_B for each $b_x^i \in B^*$ with $x \in V_j$. It is easy to see that $I_B = \{(i, j) \mid 1 \leq i, j \leq k\}$. Similarly, we also define the multiset I_H containing k^2 pairs of indices, obtained by putting (i, j) into I_H for each $h_x^i \in H^*$ with $x \in V_j$. By the previous paragraph, I_H can be obtained from I_B by taking some pair (i, j) from I_B and replacing them with corresponding pairs (i', j) where $i' > i$. Let the *measure* of a multiset of pairs I be $\mu(I) = \sum_{(i,j) \in I} i + j$. Then, $\mu(I_H) \geq \mu(I_B) = k^2(k+1)$.

By the above arguments, if for some $i < j$ the pair (i, j) is contained with multiplicity m_1 in I_H, and (j, i) is contained with multiplicity m_2 in I_H, then the candidate $m^{i,j}$ has to send $m_1 + m_2$ points to r. Similarly, if (i, i) is contained in I_H with multiplicity m, then $m^{i,i}$ has to send m points to r. Thus, $\mu(I_H)$ equals the value obtained by summing up $i + j$ for each $m^{i,j}$ and for each point transferred from $m^{i,j}$ to r. However, each $m^{i,j}$ (where $i < j$) can only send two points to r, and each $m^{i,i}$ can only send one point to r, implying $\mu(I_H) \leq \sum_{i \in [k]}(i + i) + 2\sum_{1 \leq i < j \leq k}(i + j) = k^2(k+1) = \mu(I_B)$. Hence, the measures of I_B and I_H must be equal, from which $I_H = I_B$ follows. Thus, only case B can happen.

Therefore, Γ must send one point from b_x^i to $\widetilde{c}_x^i$ at a cost of 2, and apply three more swaps of cost 3 to transfer one point from $\widetilde{c}_x^i$ to f_x^i. But in the case $i \geq 2$, this can only be done avoiding any swap of cost $1 + \varepsilon$ in the vote $w_S(i, x)$, if f_x^{i-1} simultaneously receives one point from c_x^{i-1} in $w_S(i, x)$ as well, which implies $b_x^{i-1} \in B^*$. Applying this argument iteratively, this shows that $b_x^i \in B^*$ implies $\{b_x^h \mid h < i\} \subseteq B^*$. Hence, B^* is the union of k sets of the form $\{h_x^1, h_x^2, \ldots, h_x^k\}$, implying $\sigma(1, j) = \sigma(2, j) = \cdots = \sigma(k, j)$ for each $j \in [k]$.

Finally, consider the swaps that transfer one point from $h_x^i \in H^*$ to $m^{i,j}$ in W_C where $x \in V_j$ and $i < j$. We know that if $x \in V_j$, then this must be done by applying some swaps in the vote $w_C(i, j, y, x)$ for some $y \in V_i$ such that $xy \in E$. But because of our budget, each such swap must have cost 1 and not $1 + \varepsilon$, which can only happen if Γ transforms $w_C(i, j, y, x) = (h_x^i, \widetilde{h}_y^j, \widetilde{m}^{i,j}, m^{i,j}, \dagger)$ into $(\widetilde{m}^{i,j}, m^{i,j}, h_x^i, \widetilde{h}_y^j, \dagger)$. But this implies that h_y^j must also be in B^*, implying $y = \sigma(j, i)$. Therefore we obtain that $\sigma(i, j)$ and $\sigma(j, i)$ must be vertices connected by an edge in $\mathcal{G}$. This proves the existence of a k-clique in $\mathcal{G}$, proving the theorem. □

Looking into the proof of Theorem 3, we can see that the results hold even if the costs are uniform in the sense that swapping two given candidates has the same price in any vote, and the maximum number of swaps allowed in a vote is at most 4.

By applying minor modifications to the given reduction, Theorem 3 can be generalized to hold for the case when we want p to be the unique winner; we only have to set $\text{score}(p, W) = K + 1$ in the construction. Also, Theorem 3 remains true for the case of k-approval for some $3 \leq k \leq |C| - 2$ instead of 2-approval: to prove this, it suffices to insert $k - 2$ dummies into the first $k - 2$ positions of each vote.

4 Other parameterizations

In this section, we will consider different kinds of parameterizations. First, we will look at the parameter 'number of candidates'. For this case, the following observation is helpful.

Let $S_m = \{\pi_1, \pi_2, \ldots, \pi_{m!}\}$ be the set of permutations of size m. We say that an election system is *described by linear inequalities*, if for a given set $C = \{c_1, c_2, \ldots, c_m\}$ of candidates it can be characterized by $f(m)$ sets $A_1, A_2, \ldots A_{f(m)}$ (for some computable function f) of linear inequalities over $m!$ variables $x_1, x_2, \ldots, x_{m!}$ in the following sense: if n_i denotes the number of those votes in a given election E that order C according to π_i, then the first candidate c_1 is a winner of the election if and only if for at least one index i, the setting $x_j = n_j$ for each j satisfies all inequalities in A_i.

It is easy to see that many election systems can be described by linear inequalities: any system based on scoring rules, Copeland$^\alpha$ ($0 \leq \alpha \leq 1$), Maximin, Bucklin, Ranked pairs. For example, k-approval is described by the following set A_1 of linear inequalities:

$$A_1: \quad \sum_{i:\mathrm{rank}(c_1,v_i)\leq k} x_i \geq \sum_{i:\mathrm{rank}(c_j,v_i)\leq k} x_i \quad \text{for each } 2 \leq j \leq m.$$

Theorem 4. Swap Bribery *is FPT if the parameter is the number of candidates, for any election system described by linear inequalities.*

Proof. Let $C = \{c_1, c_2, \ldots, c_m\}$ be the set of candidates, where c_1 is the preferred one, and let $A_1, A_2, \ldots A_{f(m)}$ be the sets of linear inequalities over variables $x_1, \ldots, x_{m!}$ describing the given election system $\mathcal{E}$. For some $\pi_i \in S_m$, let v_i denote the vote that ranks C according to π_i. We describe the set V of votes by writing n_i for the multiplicity of the vote v_i in V.

Our algorithm solves $f(m)$ integer linear programs with variables $T = \{t_{i,j} \mid i \neq j, 1 \leq i,j \leq m!\}$. We will use $t_{i,j}$ to denote the number of votes v_i that we transform into votes v_j; we will require $t_{i,j} \geq 0$ for each $i \neq j$. Let V^T denote the set of votes obtained by transforming the votes in V according to the variables $t_{i,j}$ for each $i \neq j$. Such a transformation from V is feasible if $\sum_{j\neq i} t_{i,j} \leq n_i$ holds for each $i \in [m!]$ (inequality $\mathcal{A}$). By [9], we can compute the price $c_{i,j}$ of transforming the vote v_i into v_j in $O(m^3)$ time. Transforming V into V^T can be done with total cost at most β, if $\sum_{i,j\in[m!]} t_{i,j}c_{i,j} \leq \beta$ (inequality $\mathcal{B}$).

We can express the multiplicity x'_i of the vote v_i in V^T as $x'_i = n_i + \sum_{j\neq i} t_{j,i} - \sum_{i\neq j} t_{i,j}$. For some $i \in [f(m)]$, let A'_i denote the set of linear inequalities over the variables in T that are obtained from the linear inequalities in A_i by substituting x_i with the above given expression for x'_i. Using the description of $\mathcal{E}$ with the given linear inequalitites, we know that the preferred candidate c_1 wins in the $\mathcal{E}$-election (C, V^T) for some values of the variables $t_{i,j}$ if and only if these values satisfy the inequalities of A'_i for at least one $i \in [f(m)]$. Thus, our algorithm solves Swap Bribery by finding a non-negative assignment for the variables in T that satisfies both the inequalities $\mathcal{A}$, $\mathcal{B}$, and all inequalities in A'_i for some i.

Solving such a system of linear inequalities can be done in linear FPT time, if the parameter is the number of variables [17]. By $|T| = (m! - 1)m!$ the theorem follows. ☐

Similarly, we can also show fixed-parameter tractability for other problems if the parameter is the number of candidates, e.g. for Possible Winner (this was already obtained by Betzler et al. for several voting systems, [3]), Manipulation (both for weighted and unweighted voters), several variants of Control (this result was obtained for Llull and Copeland voting by Faliszewski et al., [12]), or Lobbying [6] (here, the parameter would be the number of issues in the election). Since our topic is Swap Bribery, we omit the details.

Finally, we consider a combined parameter and obtain fixed-parameter tractability.

Theorem 5. *If the minimum cost is* 1*, then* Swap Bribery *for k-approval (where k is part of the input) with combined parameter $(|V|, \beta)$ admits a kernel with $O(|V|^2\beta)$ votes and $O(|V|^2\beta^2)$ candidates. Here, V is the set of votes and β is the budget.*

Proof. Let V, C, $p \in C$, and β denote the set of votes, the set of candidates, the preferred candidate, and the budget given, respectively. The idea of the kernelization algorithm is that not all candidates are interesting for the problem: only candidates that can be moved within the budget β from a zero-position to a one-position or vice versa are relevant.

Let Γ be a set of swaps with total cost at most β. Clearly, as the minimum possible cost of a swap is 1, we know that there are only 2β candidates c in a vote $v \in V$ for which $\mathrm{score}(c, v) \neq \mathrm{score}(c, v^\Gamma)$ is possible, namely, such a c has to fulfill $k - \beta + 1 \leq \mathrm{rank}(c, v) \leq k + \beta$. Thus, there are at most $2\beta|V|$ candidates for which $\mathrm{score}(c, V) \neq \mathrm{score}(c, V^\Gamma)$ is possible; let us denote the set of these candidates by $\widetilde{C}$. Let c^* be a candidate in $C \setminus \widetilde{C}$ whose score is the maximum among the candidates in $C \setminus \widetilde{C}$.

Note that a candidate $c \in C \setminus (\widetilde{C} \cup \{c^*, p\})$ has no effect on the answer to the problem instance. Indeed, if $\text{score}(p, V^\Gamma) \geq \text{score}(c^*, V^\Gamma)$, then the score of c is not relevant, and conversely, if $\text{score}(p, V^\Gamma) < \text{score}(c^*, V^\Gamma)$ then p loses anyway. Therefore, we can disregard each candidate in $C \setminus \widetilde{C}$ except for c^* and p.

The kernelization algorithm constructs an equivalent instance K as follows. In K, neither the budget, nor the preferred candidate will be changed. However, we will change the value of k to be $\beta+1$, so the kernel instance K will contain a $(\beta+1)$-approval election[3]. We define the set V_K of votes and the set C_K of candidates in K as follows.

First, the algorithm "truncates" each vote v, by deleting all its positions (together with the candidates in these positions) except for the 2β positions between $k-\beta+1$ and $k+\beta$. Then again, we shall make use of dummy candidates (see the proof of Theorem 3); we will ensure $\text{score}(d, V^\Gamma) \leq 1$ for each such dummy d. Swapping a dummy with any other candidate will have cost 1 in K. Now, for each obtained truncated vote, the algorithm inserts a dummy candidate in the first position, so that the obtained votes have length $2\beta+1$. In this step, the algorithm also determines the set $\widetilde{C}$ and the candidate c^*. This can be done in linear time. We denote the votes[4] obtained in this step by V_r. We do not change the costs of swapping candidates of $\widetilde{C} \cup \{c^*, p\}$ in some vote $v \in V_r$.

Next, to ensure that K is equivalent to the original instance, the algorithm constructs a set V_d of votes such that $\text{score}(c, V_r \cup V_d) = \text{score}(c, V)$ holds for each candidate c in $\widetilde{C} \cup \{p, c^*\}$. This can be done by constructing $\text{score}(c, V) - \text{score}(c, V_r)$ newly added votes where c is on the first position, and all the next 2β positions are taken by dummies. This way we ensure $\text{score}(c, V_d) = \text{score}(c, V_d^\Gamma)$ for any set Γ of swaps with total cost at most β.

If D is the set of dummy candidates created so far, then let $C_K = \widetilde{C} \cup \{p, c^*\} \cup D$. To finish the construction of the votes, it suffices to add for each vote $v \in V_r \cup V_d$ the candidates not yet contained in v, by appending them at the end (starting from the $(2\beta+1)$-th position) in an arbitrary order. The obtained votes will be the votes V_K of the kernel.

The presented construction needs polynomial time. Using the above mentioned arguments, it is straightforward to verify that the constructed kernel instance is indeed equivalent to the original one. Thus, it remains to bound the size of K.

Clearly, $|\widetilde{C} \cup \{p, c^*\}| \leq 2|V|\beta + 2$. The number of dummies introduced in the first phase is exactly $|V_r| = |V|$. As the score of any candidate in V is at most $|V|$, the number of votes created in the second phase is at most $(2|V|\beta + 2)|V|$, which implies that the number of dummies created in this phase is at most $(2|V|\beta+2)|V| \cdot 2\beta$. This shows $|C_K| \leq |V| + (2|V|\beta+2)(2|V|\beta+1) = O(|V|^2\beta^2)$, and also $|V_K| \leq (2|V|\beta+3)|V| = O(|V|^2\beta)$. □

Applying similar ideas, a kernel with $(k+\beta)|V|$ candidates is easy to obtain, which might be favorable to the above result in cases where k is small.

5 Conclusion

We have taken the first step towards parameterized and multivariate investigations of Swap Bribery under certain voting systems. We obtained W[1]-hardness for k-approval if the parameter is the budget β, while Swap Bribery could be shown to be in FPT for a very large class of voting systems if the parameter is the number of candidates. This revaluates previous NP-hardness results: Swap Bribery could be computed efficiently if the number of candidates is small, which is a common setting, e.g. in presidential elections.

However, we have shown this via an integer linear program formulation, using a result by Lenstra, which does not provide running times that are suitable in practice. Here, it would be interesting to give combinatorial algorithms that compute an optimal swap bribery.

[3] We use $\beta+1$ instead of β to avoid complications with the case $\beta = 0$.

[4] In fact, these vectors are not real votes yet in the sense that they do not contain each candidate.

As Elkind et al. [9] pointed out, it would be nice to characterize further natural cases of Swap Bribery that are polynomial-time solvable. We provided one such example with Theorem 1 for k-approval in the case where costs are equal to 1. By contrast, as soon as we have two different costs, the problem becomes NP-complete for k-approval ($2 \leq k \leq m-2$) and W[1]-hard if the parameter is the budget β.

There are plenty of possibilities to carry on our initiations. First, there are more parameterizations to be looked at, and in particular the study of combined parameters in the spirit of Niedermeier [20], see e.g. [1], is an interesting approach.

Also, we have focused our attention to k-approval, but the same questions could be studied for other voting systems, or for the special case of Shift Bribery which was shown to be NP-complete for several voting systems [9], or other variants of the bribery problem as mentioned in the introduction. For instance, we have only looked at *constructive* swap bribery, but the case of *destructive* swap bribery (when our aim is to achieve that a disliked candidate does *not* win) is worth further investigation as well.

Acknowledgments. We thank Rolf Niedermeier for an inspiring initial discussion.

References

1. N. Betzler. On problem kernels for possible winner determination under the k-approval protocol. In *Proc. of 35th MFCS*, 2010.
2. N. Betzler and B. Dorn. Towards a dichotomy for the Possible Winner problem in elections based on scoring rules. *J. Comput. Syst. Sci.*, 76:812–836, 2010.
3. N. Betzler, S. Hemmann, and R. Niedermeier. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proc. of 21st IJCAI*, pages 53–58, 2009.
4. N. Betzler and J. Uhlmann. Parameterized complexity of candidate control in elections and related digraph problems. *Theor. Comput. Sci.*, 410(52):5425–5442, 2009.
5. S. J. Brams and P. C. Fishburn. Voting procedures. In *Handbook of Social Choice and Welfare*, volume 1, pages 173–236. Elsevier, 2002.
6. R. Christian, M. Fellows, F. Rosamond, and A. Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, November 2007.
7. V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *J. ACM*, 54(3):1–33, 2007.
8. R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.
9. E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. In *Proc. of 2nd SAGT*, volume 5814 of *LNCS*, pages 299–310. Springer, 2009.
10. P. Faliszewski. Nonuniform bribery. In *Proc. 7th AAMAS*, pages 1569–1572, 2008.
11. P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. The complexity of bribery in elections. In *Proc. of 21st AAAI*, pages 641–646, 2006.
12. P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Llull and Copeland voting computationally resist bribery and constructive control. *J. Artif. Intell. Res. (JAIR)*, 35:275–341, 2009.
13. M. R. Fellows, D. Hermelin, F. A. Rosamond, and S. Vialette. On the parameterized complexity of multiple-interval graph problems. *Theor. Comput. Sci.*, 410(1):53–61, 2009.
14. J. Flum and M. Grohe. *Parameterized Complexity Theory.* Springer, 2006.
15. E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. *J. Comput. Syst. Sci.*, 73(1):73–83, 2007.
16. K. Konczak and J. Lang. Voting procedures with incomplete preferences. In *Proc. of IJCAI-2005 Multidisciplinary Workshop on Advances in Preference Handling*, 2005.
17. H. Lenstra. Integer programming with a fixed number of variables. *Math. of OR*, 8:538–548, 1983.
18. H. Liu, H. Feng, D. Zhu, and J. Luan. Parameterized computational complexity of control problems in voting systems. *Theor. Comput. Sci.*, 410(27-29):2746–2753, 2009.
19. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.
20. R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proc. of 27th STACS*, pages 17–32, 2010.

Appendix

Proof (of Theorem 1). Let V be the set of votes and C be the set of candidates. The score of any candidate is an integer between 0 and $|V|$. Our algorithm finds out for each possible s^* with $1 \leq s^* \leq |V|$ whether there is a solution in which the preferred candidate p wins with score s^*.

Given a value s^*, we answer the above question by solving a corresponding minimum cost maximum flow problem. We will define a network $N = (G, s, t, g, w)$ on a directed graph $G = (D, E)$ with a source vertex s and a target vertex t, where g denotes the capacity function and w the cost function defined on E. First, we introduce the vertex sets $A = \{a_{v,c} \mid v \in V, c \in C, \text{rank}(c, v) \leq k\}$, $A' = \{a'_{v,c} \mid v \in V, c \in C\}$ and $B = \{b_c \mid c \in C\}$, and we set $D = \{s, t, x\} \cup A \cup A' \cup B$. We define the arcs E as the union of the sets $E_S = \{sa \mid a \in A\}$, $E_A = \{a_{v,c}a'_{v,c} \mid \text{rank}(c, v) \leq k\}$, $E_{A'} = \{a_{v,c}a'_{v,c'} \mid \text{rank}(c, v) \leq k, \text{rank}(c', v) > k\}$, $E_B = \{a'_{v,c}b_c \mid v \in V, c \in C\}$, $E_X = \{b_c x \mid c \in C, c \neq p\}$, plus the arcs $b_p t$ and xt. We set the cost function w to be 0 on each arc except for the arcs of $E_{A'}$, and we set $w(a_{v,c}a'_{v,c'}) = \text{rank}(c', v) - \text{rank}(c, v)$. We let the capacity g be 1 on the arcs of $E_S \cup E_A \cup E_{A'} \cup E_B$, we set it to be s^* on the arcs of $E_X \cup \{b_p t\}$, and we set $g(xt) = |V|k - s^*$.

The soundness of the algorithm and hence the theorem itself follows from the following observation: there is a flow of value $|V|k$ on N having total cost at most β if and only if there exists a set Γ of swaps with total cost at most β such that $\text{score}(p, V^\Gamma) = s^*$ and $\text{score}(c, V^\Gamma) \leq s^*$ for any $c \in C, c \neq p$.

First, suppose that such a flow f exists. Since all capacities and costs are integrals, we know that f is integral as well. For each vote $v \in V$, we define a set of swaps on v as follows. We define two sets $X^\rightarrow(v)$ and $X^\leftarrow(v)$ in a way that if $f(a_{v,c}a'_{v,c'}) = 1$ holds for some c and c' with $c \neq c'$, then we put c into $X^\rightarrow(v)$ and we put c' into $X^\leftarrow(v)$. Clearly, $|X^\rightarrow(v)| = |X^\leftarrow(v)|$, by the given capacities. Observe that moving the candidates in $X^\rightarrow(v)$ to the positions $k+1, k+2, \ldots, k+h$ and also the candidates in $X^\leftarrow(v)$ to the positions $k, k-1, \ldots, k-h+1$ for $h = |X^\rightarrow(v)|$ has total cost $\sum_{c' \in X^\leftarrow(v)} \text{rank}(c', v) - \sum_{c \in X^\rightarrow(v)} \text{rank}(c, v)$. Thus, by letting $\Gamma(v)$ contain these swaps for some v, we know that the cost of the bribery $\Gamma = \{\Gamma(v) \mid v \in V\}$ is exactly the cost of the flow f which is not more than β. Observe that as a result of these swaps, each candidate c other than p will receive at most s^* scores in V^Γ because of the capacity $g(b_c x) \leq s^*$. On the other hand, by $g(xt) = |V|k - s^*$ we get $f(b_p t) = s^*$, which yields that p will receive exactly s^* scores in V^Γ. Thus, Γ has the properties claimed.

For the converse direction, let Γ be a set of swaps with total cost at most β such that $\text{score}(p, V^\Gamma) = s^*$ and $\text{score}(c, V^\Gamma) \leq s^*$ for any $c \in C, c \neq p$. For some $v \in V$, let $X^\rightarrow(v)$ denote those candidates c for which $\text{score}(c, v) > \text{score}(c, v^\Gamma)$, and let $X^\leftarrow(v)$ denote those candidates c' for which $\text{score}(c', v) < \text{score}(c', v^\Gamma)$. It is easy to see that the swaps applied in v by Γ have total cost at least $\sum_{c \in X^\leftarrow(v)} \text{rank}(c', v) - \sum_{c \in X^\rightarrow(v)} \text{rank}(c, v)$. Therefore, a flow can be easily constructed having cost at most β in the following way: for each v and c where $\text{score}(c, v) = \text{score}(c, v^\Gamma) = 1$ we let $f(a_{v,c}a'_{v,c}) = 1$, and for each v and c where c is the i-th candidate in $X^\rightarrow(v)$ we set $f(a_{v,c}a'_{v,c'}) = 1$ for the i-th candidate C' of $X^\leftarrow(v)$ according to some fixed ordering. It is not hard to verify that this indeed determines a flow for N, with value $|V|k$ and cost at most β. □

Proof (of Proposition 2). It has already been proved by Elkind et al. [9] that Possible Winner reduces to Swap Bribery if the possible costs include 0 and 1, and the budget is zero. Clearly, the result also holds if we assume that the costs include 0 and δ for some $\delta > 0$.

For the other direction, it is easy to see that a Swap Bribery instance with costs in $\{0, \delta\}$, $\delta > 0$ and budget zero is equivalent to the Possible Winner instance with the same candidates where each vote v is replaced by the transitive closure of the relation $\prec_v$ for which $a \prec_v b$ holds if and only if a precedes b in the vote v and the cost of swapping a with b in v is non-zero. $\square$