Swap Bribery

Edith Elkind[1,2], Piotr Faliszewski[3], and Arkadii Slinko[4]

[1] School of ECS, University of Southampton, UK
[2] Division of Mathematical Sciences, Nanyang Technological University, Singapore
[3] Dept. of Computer Science, AGH Univ. of Science and Technology, Kraków, Poland
[4] Dept. of Mathematics, University of Auckland, New Zealand

Abstract. In voting theory, *bribery* is a form of manipulative behavior in which an external actor (the briber) offers to pay the voters to change their votes in order to get her preferred candidate elected. We investigate a model of bribery where the price of each vote depends on the amount of change that the voter is asked to implement. Specifically, in our model the briber can change a voter's preference list by paying for a sequence of swaps of consecutive candidates. Each swap may have a different price; the price of a bribery is the sum of the prices of all swaps that it involves. We prove complexity results for this model, which we call *swap bribery*, for a broad class of voting rules, including variants of approval and k-approval, Borda, Copeland, and maximin.

1 Introduction

There is a range of situations in social choice where an external actor may alter some of the already submitted votes, or the votes that the voters intend to submit. For example, a candidate can attempt to change the voters' preferences by running a campaign, which may be targeted at a particular group of voters. A more extreme (and illegal) version of this strategy involves paying voters to change their votes, or bribing election officials to get access to already submitted ballots in order to modify them. Alternatively, one can assume that the submitted votes can be contaminated with random mistakes, and a central authority should be allowed to correct the votes (preferably, by changing them as little as possible) to reveal the true winner. Indeed, this scenario is, in fact, one of the original motivations behind Dodgson's voting rule. (See also papers [16, 6].)

All of these activities can be interpreted as changing the voters' preferences subject to a budget constraint, and can therefore be studied using the notion of bribery in elections introduced by Faliszewski, Hemaspaandra, and Hemaspaandra [10]. In their model of bribery, we are given an election (i.e., a set of candidates and a list of votes), a preferred candidate p, a price of each vote, and a budget B. We ask if there is a way to pick a group of voters whose total price is at most B so that via changing their votes we can make p a winner.

In the model of Faliszewski, Hemaspaandra, and Hemaspaandra [10] each voter may have a different price, but this price is fixed and does not depend on the nature of the requested change: upon paying a voter, the briber can modify her vote in any way. While there are natural scenarios captured by this model, it fails to express the fact that voters may be more willing to make a small change to their vote (e.g., swap their 2nd

and 3rd most favorite candidates) than to change it completely. To account for such settings, Faliszewski [9] proposed a new notion of bribery, which he called *nonuniform bribery*. Under nonuniform bribery, a voter's price may depend on the nature of changes she is asked to implement. A similar notion called *microbribery* was considered in [11]. However, none of these papers considers the standard model of elections, in which votes are preference orders over the set of candidates. Specifically, Faliszewski [9] focused on the so-called utility-based voting, while Faliszewski et al. [11] used the irrational voter model, in which voters' preferences may contain cycles.

The goal of this paper is to study a notion of nonuniform bribery that can be used within the standard model of elections. Our framework, which we call *swap bribery*, is inspired by Dodgson voting rule (see Fellows, Rosamond, and Slinko [12] for a related discussion). We use the name "swap bribery" as it precisely captures the nature of our framework. Specifically, in swap bribery, the briber can ask a voter to perform a sequence of swaps; each swap changes the relative order of two candidates that are currently adjacent in this voter's preference list. For example, if a voter prefers a to b and b to c (we write this as $a \succ b \succ c$), she can be asked to swap a and b, then a and c, then b and c, resulting in the vote $c \succ b \succ a$. Each swap has an associated price, and the total price is simply the sum of the prices of individual swaps. When preferences are viewed as orderings, a swap of adjacent candidates is a natural "atomic" operation on a vote. Moreover, one can transform any vote into any other vote by a sequence of such swaps. Hence, attaching prices to such operations provides a good model for nonuniform bribery in the standard setting.

We also study a special case of swap bribery, which we call *shift bribery*. Under this model of bribery the only allowable swaps are the ones that involve the preferred candidate. Thus, in effect, a shift bribery amounts to asking a voter to move the preferred candidate up by a certain number of positions in her preference order. As argued above, bribery can be used to model a legitimate approach to influencing elections, namely, campaigning: the "briber" simply invests money into trying to convince a particular group of voters that one candidate is better than another. The message and costs of the campaign can vary from one group of voters to another, which is captured by different bribery prices. In this context, shift bribery corresponds to campaigning for the preferred candidate (as opposed to discussing relative merits of other candidates), and is therefore particularly appealing.

After introducing our model of bribery, we proceed to study it from the algorithmic perspective. Our goal here is threefold. First, as argued above, despite its negative connotations, bribery may correspond to perfectly legal and even desirable behavior, and therefore we are interested in developing efficient algorithms that a potential "briber" (that is, a campaign manager) can use. Second, from a more technical perspective, we would like to pinpoint the source of computational hardness in nonuniform bribery. Indeed, when the number of candidates is unbounded, the general bribery of Faliszewski, Hemaspaandra, and Hemaspaandra [10] appears to be hard for all but the simplest voting rules. In contrast, there is a number of polynomial-time algorithms for nonuniform bribery in non-standard models, such as utility-based voting or irrational voters. We would like to know whether these easiness results are tied to the increased flexibility of pricing in nonuniform bribery, or to the increased flexibility of the alternative voter

models. The results of this paper, most of which are NP-completeness results, suggest that the latter is true. We are also motivated by the "computational hardness as a barrier against manipulation" line of work, pioneered by Bartholdi, Tovey, and Trick [1]. While it has since been argued that NP-hardness might not provide sufficient protection against dishonest behavior and that more robust notions of hardness are needed (see, e.g., [21, 13, 19, 20]), identifying settings in which bribery is NP-hard is a useful first step towards finding a voting rule that is truly resistant to dishonest behavior.

This paper is organized as follows. After providing the necessary background in Section 2, in Section 3 we formally define our model of bribery, and prove some general results about swap bribery. Section 4 contains a detailed study of bribery in approval voting. In Section 5, we consider other popular voting rules, such as Borda, Copeland, and maximin. We conclude with several directions for further research in Section 6. We omit most of the proofs due to space constraints; these proofs appear in the full version of the paper [7].

2 Preliminaries

Elections. An *election* is a pair $E = (C, V)$, where $C = \{c_1, \ldots, c_m\}$ is a set of *candidates* and $V = (v_1, \ldots, v_n)$ is a list of *voters*. Each voter v_i is represented via her *preference order* $\succ_i$, which is a strict linear order over the candidates in C (in the context of the possible-winner problem we also allow partial orders). For example, given $C = \{c_1, c_2, c_3\}$ and $V = (v_1, v_2)$, we write $c_2 \succ_2 c_1 \succ_2 c_3$ to denote that the second voter, v_2, prefers c_2 to c_1 to c_3. For any $C' \subseteq C$, by writing C' in a preference order we mean listing all the elements of C' in an arbitrary but fixed order. Similarly, $\overleftarrow{C'}$ means listing members of C' in the reverse of this fixed order.

A *voting rule* $\mathcal{E}$ maps an election $E = (C, V)$ to a set $W \subseteq C$ of *winners*. We assume the nonunique-winner model: all members of $\mathcal{E}(E)$ are considered to be winning. All voting rules considered in this paper are point-based: they assign, via some algorithm, points to candidates, and declare as winners the ones with most points. For an election $E = (C, V)$, we denote by $\mathrm{score}_E(c_i)$ the number of points that a candidate $c_i \in C$ receives in E according to a given voting rule. Sometimes, to disambiguate, we will indicate in the superscript the particular voting rule used. We will provide the definitions of the relevant rules as we discuss them in further sections.

Manipulation, Possible Winners, and Bribery. In this paper we take *manipulation* to mean unweighted constructive coalitional manipulation as defined by Conitzer, Lang, and Sandholm [5]. That is, in $\mathcal{E}$-manipulation we are given an election $E = (C, V)$, a preferred candidate p, and a list of "manipulative" voters V', and we ask if it is possible to set the preferences of voters in V' so that p is an $\mathcal{E}$-winner of $(C, V \cup V')$. In the $\mathcal{E}$-possible-winner problem we are given an election $E = (C, V)$, where the voters' preference are (possibly) *partial*, i.e., are given by partial orders over C, and we ask if it is possible to *complete* the votes so that a given candidate p is an $\mathcal{E}$-winner of the resulting election. It is not hard to see that $\mathcal{E}$-manipulation is a special case of $\mathcal{E}$-possible-winner where some votes are completely specified and some (i.e., those of the manipulative voters) are completely unspecified . The study of possible-winner problems was initiated by Konczak and Lang [15] and then continued by multiple other

authors (see, e.g., Walsh's overview paper [17] and the work of Xia and Conitzer [18]). Finally, in $\mathcal{E}$-bribery [10], we are given an election $E = (C, V)$, a preferred candidate p, a list of voters' prices and a nonnegative integer B, and we ask if it is possible to modify votes at a total cost of at most B so that p becomes an $\mathcal{E}$-winner of the resulting election. (In [10] "bribery" refers to the case where all voters have unit prices, and the more general setting described above is called $bribery.)

Computational Complexity. We assume familiarity with standard notions of computational complexity such as the classes P and NP, NP-completeness, and (polynomial-time) many-one reductions. Many of our hardness proofs rely on reductions from the NP-complete problem EXACT COVER BY 3-SETS (X3C) [14].

Definition 1 ([14]). *An instance $(\mathcal{B}, \mathcal{S})$ of* EXACT COVER BY 3-SETS (X3C) *is given by a ground set $\mathcal{B} = \{b_1, \ldots, b_{3K}\}$, and a family $\mathcal{S} = \{S_1, \ldots, S_M\}$ of subsets of $\mathcal{B}$, where $|S_i| = 3$ for each $i = 1, \ldots, M$. It is a "yes"-instance if there is a subfamily $\mathcal{S}' \subseteq \mathcal{S}$, $|\mathcal{S}'| = K$, such that for each $b_i \in \mathcal{B}$ there is an $S_j \in \mathcal{S}'$ such that $b_i \in S_j$, and a "no"-instance otherwise.*

3 Swap Bribery

In any reasonable model of nonuniform bribery, one should be able to specify the price for getting a given voter to submit any preference ordering (some of these orderings may be unacceptable to the voter, in which case the corresponding price should be set to $+\infty$). However, in elections with m candidates, there are $m!$ possible votes, so listing the prices of these votes explicitly is not practical. Alternatively, one could specify the bribery prices via an oracle, i.e., via a polynomial-time algorithm that, given a voter i and a preference order $\succ$, outputs the price for getting i to vote according to $\succ$. However, without any restrictions on the oracle, even finding a cheapest way to affect a given vote will require exponentially many queries, and providing appropriate restrictions would be challenging.

Against this background, we will now present a model of bribery that allows for easy specification of bribery prices, and yet is expressive enough to capture many interesting scenarios. Our model is based on the following idea. Intuitively, an atomic operation on a given vote is a swap of two consecutive candidates. Moreover, one can transform any vote into any other vote by a sequence of such steps. It is therefore natural to assume that the price for such transformation is reasonably well approximated by the sum of the prices of individual swaps. We now proceed to formalize this approach.

Let $E = (C, V)$ be an election, where $C = \{c_1, \ldots, c_m\}$ and $V = (v_1, \ldots v_n)$. A *swap-bribery price function* is a mapping $\pi \colon C \times C \to \mathbb{N}$, which for any ordered pair of candidates (c_i, c_j) specifies a number $\pi(c_i, c_j)$, Intuitively, this number is the price of swapping c_i and c_j in a given voter's preference order. More precisely, for a voter v_k with a swap-bribery price function π_k, a *unit swap* is a triple (v_k, c_i, c_j). A unit swap is *admissible* if c_i immediately precedes c_j in v_k's preference order; its price is $\pi_k(c_i, c_j)$. Executing an admissible unit swap (v_k, c_i, c_j) means changing v_k's preference order from $\ldots \succ c_i \succ c_j \succ \ldots$ to $\ldots \succ c_j \succ c_i \succ \ldots$.

Note that we do not allow swapping non-adjacent candidates in a single step (though, of course, such a swap could be simulated by a sequence of swaps of adjacent candidates). Indeed, such a swap would change these candidates' order relative to all candidates that appear between them in the vote.

Definition 2. *For any voting rule $\mathcal{E}$, an instance of $\mathcal{E}$-*swap-bribery *is given by an election $E = (C, V)$ with $C = \{c_1, \ldots, c_m\}$, $p = c_1$ and $V = (v_1, \ldots, v_n)$, a list of voters' swap-bribery price functions $(\pi_1, \ldots, \pi_n)$, and a nonnegative integer B (the budget). We ask if there exists a sequence $(s_1, \ldots, s_t)$ of unit swaps such that (1) when executed in order, each unit swap is admissible at the time of its execution, (2) executing $s_1, \ldots, s_t$ ensures that p is a winner of the resulting $\mathcal{E}$-election, and (3) the sum of the prices of executing $s_1, \ldots, s_t$ is at most B.*

As argued above, swap bribery can be used to transform any vote into any other vote. It is natural to ask if one can efficiently compute an optimal way of doing so. It turns out that the answer to this question is "yes".

Proposition 1. *Given two votes $v_1 = c_{i_1} \succ_1 \ldots \succ_1 c_{i_m}$ and $v_2 = c_{j_1} \succ_2 \ldots \succ_2 c_{j_m}$, and a swap-bribery price function π, one can compute in polynomial time the cheapest (with respect to π) sequence of swaps converting v_1 into v_2.*

Proof. Set $\mathcal{I}(v_1, v_2) = \{(c_i, c_j) \mid c_i \succ_1 c_j, c_j \succ_2 c_i\}$; we say that a pair of candidates $(c_i, c_j) \in \mathcal{I}(v_1, v_2)$ is *inverted*. Clearly, to obtain v_2 from v_1, it is necessary to swap each inverted pair, so the total cost of an optimal bribery is at least $s = \sum_{(c_i,c_j)\in\mathcal{I}(v_1,v_2)} \pi(c_i, c_j)$. We will now argue that one never needs to swap a pair not in $\mathcal{I}(v_1, v_2)$, or to swap a pair in $\mathcal{I}(v_1, v_2)$ more than once; this implies that the cost of an optimal bribery is exactly s.

Our argument is by induction on the size of $\mathcal{I}(v_1, v_2)$. If $|\mathcal{I}(v_1, v_2)| = 0$, then $v_1 = v_2$ and the statement is obvious. Now, suppose that the statement has been proved for all v_1', v_2' with $|\mathcal{I}(v_1', v_2')| < k$, and consider a pair (v_1, v_2) with $|\mathcal{I}(v_1, v_2)| = k$. We claim that there is a pair of candidates $(c_i, c_j) \in \mathcal{I}(v_1, v_2)$ that is adjacent in v_1. Indeed, suppose otherwise, and let (c_i, c_j) be a pair in $\mathcal{I}(v_1, v_2)$ that is the closest in v_1. By our assumption, there exists at least one $c \in C$ such that $c_i \succ_1 c \succ_1 c_j$, yet $(c_i, c) \notin \mathcal{I}(v_1, v_2)$, $(c, c_j) \notin \mathcal{I}(v_1, v_2)$. Hence, we have $c_i \succ_2 c$, $c \succ_2 c_j$, so by transitivity of $\succ_2$ we conclude $c_i \succ_2 c_j$, a contradiction with $(c_i, c_j) \in \mathcal{I}(v_1, v_2)$. Hence, $\mathcal{I}(v_1, v_2)$ always contains an adjacent pair (c_i, c_j). By swapping c_i and c_j, we obtain a vote v_1' that satisfies $|\mathcal{I}(v_1', v_2)| = k - 1$. Note also that $\mathcal{I}(v_1', v_2) = \mathcal{I}(v_1, v_2) \setminus \{(c_i, c_j)\}$, as the relative order of all other candidates with respect to c_i and c_j did not change. We can now apply our inductive hypothesis. Note that this argument implies a polynomial-time algorithm for transforming v_1 into v_2 in s steps. □

Proposition 1 shows how to optimally convert one vote into another using swaps. We can also compute in polynomial time the cheapest way of transforming a collection of votes into any other collection of votes of the same cardinality.

Proposition 2. *Given a list of votes $V = (v_1, \ldots, v_n)$, a corresponding list of price functions $(\pi_1, \ldots, \pi_n)$, and a multiset of votes $V' = \{v_1', \ldots, v_n'\}$, one can find in polynomial time an optimal swap bribery that transforms V into V'.*

The idea of the proof is to find a minimum-cost perfect matching between V and V', where the cost of each edge (v, v') is given by the price of transforming v into v' via swap bribery.

A voting rule is called *anonymous* if its outcome does not depend on the order of votes in V. Typical voting rules are anonymous. For such rules, Proposition 2 suggests a polynomial-time algorithm for finding an optimal swap bribery in the important special case where the number of candidates is fixed.

Theorem 1. *For any anonymous voting rule with a polynomial-time winner determination procedure, one can compute an optimal swap bribery in polynomial time if the number of candidates is bounded by a constant.*

The idea of the proof is to consider all possible multisets of votes that the briber might request to obtain and apply Proposition 2 to each of them. Observe that when $|C|$ is constant, the number of different multisets of votes is polynomial in $|V|$, but the number of different lists of votes is exponential in $|V|$. This is why Proposition 2 is phrased in terms of multisets of votes rather than lists of votes.

The next result allows us to quickly derive swap-bribery hardness results from possible-winner hardness results.

Theorem 2. *For any voting rule $\mathcal{E}$, $\mathcal{E}$-possible-winner many-one reduces to $\mathcal{E}$-swap-bribery.*

Proof. An instance of the $\mathcal{E}$-possible-winner problem is a pair $\langle (C, V), p\rangle$, where V may contain partial orders and $p \in C$. We will now describe a polynomial-time algorithm that transforms $\langle (C, V), p\rangle$ into an instance of $\mathcal{E}$-swap-bribery in which p can become a winner via swap bribery of cost 0 if and only if the votes in V can be completed in such a way that p is a winner of the resulting election.

Our construction works as follows. First, for each (possibly) partial vote $\succ_k$ in V we compute a complete vote $\succ'_k$ that agrees with $\succ_k$ wherever $\succ_k$ is defined. This can easily be done via, e.g., topological sorting. Next, for each vote $\succ'_k$ we construct a price function π_k as follows. For any pair of candidates $c_i, c_j \in C$, we set $\pi_k(c_i, c_j) = 1$ if $c_i \succ_k c_j$ and $\pi_k(c_i, c_j) = 0$ otherwise. We output an instance of swap bribery with budget 0, preferred candidate p and an election E' which is identical to E except that each vote $\succ_k$ is replaced by vote $\succ'_k$ associated with price function π_k.

Clearly, this reduction works in polynomial time. To prove its correctness, fix an index k and consider a vote $\succ'_k$ and an arbitrary vote $\succ''_k$. We claim that $\succ'_k$ can be transformed into $\succ''_k$ via a swap bribery of cost 0 (with respect to π_k) if and only if $\succ''_k$ agrees with $\succ_k$ on all pairs of candidates comparable under $\succ_k$. Indeed, as shown in the proof of Proposition 1, the optimal swap bribery that transforms $\succ'_k$ into $\succ''_k$ swaps each pair of candidates c_i, c_j such that $c_i \succ'_k c_j$ and $c_j \succ''_k c_i$ exactly once. Clearly, the cost of these swaps is 0 if and only if $\succ''_k$ agrees with $\succ_k$ on all pairs of candidates comparable under $\succ_k$. Consequently, the votes in E can be completed so as to make p a winner if and only if there is a swap bribery of cost 0 that makes p a winner in E'. □

Since $\mathcal{E}$-manipulation is a special case of $\mathcal{E}$-possible-winner, as a corollary we immediately obtain that $\mathcal{E}$-manipulation many-one reduces to $\mathcal{E}$-swap-bribery.

Shift bribery. In some settings, the briber may be unable to ask voters to make a swap that does not involve the preferred candidate. For example, in an election campaign investing money to support another candidate may be viewed as unethical. In such cases, the only action available to the briber is to ask a voter to move the preferred candidate up in her preference order. We will refer to this type of bribery as *shift bribery*.

Fix an election $E = (C, V)$ with $C = \{c_1, \ldots, c_m\}$, $p = c_1$, and a voter $v \in V$ with a preference order $\succ$. Suppose that p appears in the jth position in $\succ$. We say that a mapping $\rho : \mathbb{N} \to \mathbb{N}$ is a *shift-bribery price function* for v if it satisfies (1) $\rho(0) = 0$; (2) $\rho(i) \leq \rho(i')$ for $i < i' < j$; and (3) $\rho(i) = +\infty$ for $i \geq j$. We interpret $\rho(i)$ as the price of moving p up by i positions in $\succ$.

Definition 3. *For any voting rule $\mathcal{E}$, an instance of* $\mathcal{E}$-shift-bribery *is given by an election $E = (C, V)$ with $C = \{c_1, \ldots, c_m\}$, $p = c_1$ and $V = (v_1, \ldots, v_n)$, a list of voters' shift-bribery price functions $(\rho_1, \ldots, \rho_n)$, and a nonnegative integer B (the budget). We ask if there is a sequence $(k_1, \ldots, k_n)$ of nonnegative integers such that $\sum_{i=1}^{n} \rho_i(k_i) \leq B$ and bribing each voter v_i to shift p up by k_i places ensures that p is a $\mathcal{E}$-winner of the resulting election.*

It is not hard to see that $\mathcal{E}$-shift-bribery is a special case of $\mathcal{E}$-swap-bribery.

Proposition 3. *For any voting rule $\mathcal{E}$, any election $E = (C, V)$ with $C = \{c_1, \ldots, c_m\}$, $p = c_1$ and $V = (v_1, \ldots, v_n)$, and any list $(\rho_1, \ldots, \rho_n)$ of shift-bribery price functions for V, we can efficiently construct a list $(\pi_1, \ldots, \pi_n)$ of swap-bribery price functions for V so that the problem of $\mathcal{E}$-shift-bribery with respect to $(\rho_1, \ldots, \rho_n)$ is equivalent to the problem of $\mathcal{E}$-swap bribery with respect to $(\pi_1, \ldots, \pi_n)$.*

Proof. The general idea of the proof is as follows. We set the budget in the swap bribery problem to be the same as in the input shift bribery problem. To construct a swap-bribery price function π_i for a voter v_i, we renumber the candidates in C so that $c_1 = p$ and v_i's preference order is $c_k \succ_i c_{k-1} \succ_i \cdots \succ_i c_2 \succ_i p \succ_i \cdots$. Now set

$$\pi_i(x, y) = \begin{cases} \rho_i(1) & \text{if } x = p \text{ and } y = c_2 \\ \rho_i(\ell - 1) - \rho_i(\ell - 2) & \text{if } x = p \text{ and } y = c_\ell,\ \ell = 3, \ldots, k \\ +\infty & \text{in all other cases.} \end{cases}$$

A simple inductive proof shows that setting all π_i in this way proves the theorem. □

The analog of Theorem 2 does not seem to hold for shift bribery. Hence, unlike in the case of swap bribery, it is of interest to explore the complexity of shift bribery even when the corresponding possible-winner problem is known to be hard. Another natural question in this context is whether there are voting rules for which shift bribery is strictly easier than swap bribery. As our subsequent results show, the answer to this question is "yes" (assuming $\mathrm{P} \neq \mathrm{NP}$).

4 Case Study: Approval Voting

In this section we investigate the complexity of swap bribery in k-approval voting. The family of k-approval voting rules (for various values of k) is a simple but interesting class of voting rules, including such well-known rules as plurality and veto. In

k-approval, a voter assigns a point to each of the top k candidates on her preference list. Thus, 1-approval is simply the *plurality* rule and, for $|C| = m$, $(m-1)$-approval is the *veto* rule, where, in effect, each voter votes against her least desirable candidate. Our first result is that swap bribery is easy for plurality and veto but hard for almost all variants of k-approval with fixed k.

Theorem 3. *Swap bribery is in* P *for plurality (i.e.,* 1*-approval) and veto (i.e.,* $(m-1)$*-approval). However, for each fixed* k *such that* $k \geq 3$*, swap bribery for* k*-approval is* NP*-complete, even if all swaps have costs in the set* $\{0, 1, 2\}$.

We omit the proof of this theorem due to space constraints. Note that Theorem 3 does not say anything about the complexity of swap bribery for 2-approval. Very recently, Betzler and Dorn [2] have shown that for 2-approval the possible winner problem is NP-hard, and thus by Theorem 2 swap bribery for 2-approval is NP-hard as well.

In contrast to Theorem 3, shift bribery for k-approval is easy for all values of k. Thus, shift bribery can indeed be easier than swap bribery.

Theorem 4. *Shift bribery for* k*-approval is in* P *for any* $k < m$.

The proof relies on the fact that under shift bribery, the only reasonable way to bribe a given voter is to ask him to approve of p at the lowest possible cost.

Now, the NP-hardness proof in Theorem 3 assumes that both the number of candidates and the number of voters are parts of the input (i.e., are not bounded by any fixed constant). We have seen that the first requirement is necessary: by Theorem 1, swap bribery becomes easy if the number of candidates is constant. It is therefore natural to ask if the number of voters plays a similar role. It turns out that if k is bounded by a constant, swap bribery is indeed easy for each fixed number of voters.

Theorem 5. *For each fixed* k*, swap bribery for* k*-approval is in* P *if the number of voters is bounded by a constant.*

Proof. Consider an election $E = (C, V)$ with $C = \{c_1, \ldots, c_m\}$, $V = (v_1, \ldots, v_n)$, a preferred candidate $p \in C$, a budget B. and a list of price functions $(\pi_1, \ldots, \pi_n)$. Let $C_1, \ldots, C_T$ be the list of all k-element subsets of C; note that $T = \binom{m}{k} = \text{poly}(m)$. For a given vote v, we can compute the cost of moving the candidates from a given k-element subset C_t into top k positions in v. Indeed, suppose that $C_t = \{c_{i_1}, \ldots, c_{i_k}\}$, and c_{i_1} is the first of these candidates to appear in v, c_{i_2} is second, etc. Then this cost is simply the cost of moving c_{i_1} into the top position by successively swapping it with all candidates that are above him, followed by moving c_{i_2} into the second position, etc. To see why this naive algorithm is optimal, note that it only swaps pairs that are inverted in the sense of Proposition 1, i.e., ones that have to be swapped anyway.

We can now go over all lists of the form $(C_{i_1}, \ldots, C_{i_n})$, $i_j \in \{1, \ldots, T\}$ for $j = 1, \ldots, n$, and for each such list compute the cost of the optimal bribery that for $j = 1, \ldots, n$ transforms the jth input vote into a vote that lists the candidates in C_{i_j} in the top k positions. There are at most $\binom{m}{k}^n = \text{poly}(m)$ such lists; we accept if at least one of them costs at most B and bribing the voters to implement it ensures p's victory. □

On the other hand, when k is unbounded, swap bribery becomes difficult even if there is just one voter. To prove this result, we reduce from the NP-complete problem BALANCED BICLIQUE (BB) (see [14]).

Definition 4 ([14]). *An instance of* BB *is given by a bipartite graph $G = (U, W, E)$, where $|U| = |W| = N$ and $E \subseteq U \times W$, and a natural number $K \leq N$. It is a "yes"-instance if there are sets $U' \subseteq U$ and $W' \subseteq W$ such that $|U'| = |W'| = K$ and for all $u \in U'$, $w \in W'$ we have $(u, w) \in E$, and a "no"-instance otherwise.*

Intuitively, the reason why swap bribery for k-approval is difficult for large values of k is that it may be beneficial for the briber to move around some candidates other than p, as this may enable him to promote p via swaps of lower cost.

Theorem 6. *When k is a part of the input, swap bribery for k-approval is* NP*-complete even for a single voter.*

Proof. It is easy to see that our problem is in NP. We focus on the NP-hardness proof. We give a reduction from BB (see Definition 4 above). Suppose that we are given an instance of BB with $U = \{u_1, \ldots, u_N\}$, $W = \{w_1, \ldots, w_N\}$. Our election will have $2N + 1$ candidates $u_1, \ldots, u_N, w_1, \ldots, w_N, p$, where p is the preferred candidate, and a single voter v with preference ordering $U \succ W \succ p$. The price function is given by $\pi(u_i, u_j) = 0$, $\pi(w_i, w_j) = 0$ for all $i, j = 1, \ldots, N$, $\pi(w_i, p) = 1$, $\pi(u_i, p) = 0$ for all $i = 1, \ldots, N$, $\pi(u_i, w_j) = 0$ if $(u_i, w_j) \in E$ and $\pi(u_i, w_j) = N - K + 1$ otherwise. Finally, we set $k = N + 1$ and $B = N - K$.

Suppose that we have a "yes"-instance of BB, and let (U', W') be the corresponding witness. Then we can first reorder U and W for free so that $U \setminus U' \succ U'$, $W' \succ W \setminus W'$, then swap U' and W' (which is free, since (U', W') is a biclique in G), and, finally, move p past $W \setminus W'$ and U', paying $|W \setminus W'| = N - K = B$.

Conversely, suppose that there is a successful bribery for v. Let U' be the set of candidates from U that end up below p, and let W' be the set of candidates from W that end up above p after the bribery. Observe that this means that we had to swap each pair $(u, w) \in U' \times W'$, and hence $(u, w) \in E$ for all $(u, w) \in U' \times W'$, as otherwise we would have exceeded our budget. We had to pay 1 for swapping p with each of the candidates in $W \setminus W'$, so $|W \setminus W'| \leq N - K$ and hence $W' \geq K$. On the other hand, p ended up among the top $N + 1$ candidates, so $|W'| + |U \setminus U'| \leq N$, and hence $|U'| \geq K$. Pick $U'' \subseteq U'$, $W'' \subseteq W'$ so that $|U''| = |W''| = K$. The pair (U'', W'') is a balanced biclique of the required size in G because we have started with a successful bribery. □

Bribery in SP-AV. Nonuniform bribery for approval voting has already been studied thoroughly [10, 9]. Recently, Brams and Sanver [3] introduced a variant of approval voting called SP-AV, whose computational study was initiated by Erdélyi, Nowak, and Rothe [8]. In the full version of this paper [7] we discuss swap bribery for SP-AV.

5 Further Voting Rules and Shift Bribery

In this section we consider voting rules other than approval, starting with Borda. In a Borda election with m candidates, the number of points assigned by a voter v to a

candidate c equals the number of candidates that v ranks below c. The possible winner problem for Borda is NP-complete [18] and thus Theorem 2 implies that swap bribery for Borda is NP-complete. Thus, we will now focus on Borda-shift bribery.

Perhaps unsurprisingly, shift bribery for Borda turns out to be computationally hard.

Theorem 7. *Shift bribery for Borda is* NP*-complete.*

However, there exists a 2-approximation algorithm for Borda-shift bribery.

Theorem 8. *There exists a polynomial time algorithm that, given an instance $I = (C, V, p, (\rho_1, \ldots, \rho_n), B)$ of shift bribery, outputs a sequence of shifts that makes p a Borda winner, and whose cost is at most twice the cost of an optimal Borda-shift bribery for I.*

Proof. Fix an instance I of Borda-shift bribery. Suppose that the optimal shift bribery in I has cost c and moves p up by k positions in total. It is easy to see that *any* bribery in I that shifts p up by at least $2k$ positions makes p a winner. Indeed, in the optimal solution shifting p up by k positions increases p's score by k and decreases every other candidate's score by at most k. Thus, altogether the advantage that p has over any other candidate increases by at most $2k$. We obtain the same effect by shifting p up by $2k$ positions.

Suppose that we know k. Then we can use dynamic programming to compute a minimum-cost bribery that shifts p up by k positions as follows. For each $i = 1, \ldots, n$ and $k' = 1, \ldots, k$, let $f(i, k')$ be the cost of a minimum-cost shift bribery that moves p up by k' positions in the preferences of the first i voters. We have $f(1, k') = \rho_i(k')$ for $k' \leq m - k_1$, where k_1 is the position of p in the first vote, and $f(1, k') = +\infty$ for $k' > m - k_1$. Further, we have $f(i + 1, k') = \min\{f(i, k' - k'') + \rho_{i+1}(k'') \mid k'' = 1, \ldots, m - k_{i+1}\}$, where k_{i+1} is the position of p in the $(i + 1)$st vote. Denote the resulting bribery by $\mathcal{B}$. Obviously, the cost of $\mathcal{B}$ is given by $f(n, k)$, and one can compute $\mathcal{B}$ itself using standard techniques. Observe that the cost of $\mathcal{B}$ is at most c.

The bribery $\mathcal{B}$ includes some j shifts, $j \leq k$, that also appear in the optimal solution. Suppose that we know the value of j, and imagine that we first execute these j shifts. After doing so, we get an instance I' that still allows the remaining $k - j$ shifts of the optimal solution. Thus, given I', one can find $k - j$ shifts that ensure p's victory and so, by the observation in the previous paragraph, any $2(k - j)$ shifts from I' suffice to make p a winner. Let I'' be the instance obtained after executing $\mathcal{B}$. Clearly, one can transform I' into I'' using $k - j$ shifts. Therefore, in I'' any bribery that shifts p by $k - j$ positions makes p a winner. Thus, after executing $\mathcal{B}$, we pick the cheapest bribery $\mathcal{B}'$ that shifts p up by $k - j$ positions. These $k - j$ shifts cost at most c, because there are the $k - j$ unused shifts from the optimal solution, whose cost is at most c. As a result, we ensure p's victory via $2k - j$ shifts, and pay at most $2c$.

Now, the algorithm above assumes that we know k and j. When solving an arbitrary instance, we do not know them, but we can try all combinations. □

There is an interesting connection between shift bribery for Borda and computing the Dodgson score of a candidate. We point the reader to the full version of this paper [7] for a discussion comparing the algorithm described above and one of the algorithms of [4] for the Dodgson score.

We now turn to elections defined via considering majority contests between pairs of candidates. Specifically, we consider maximin and Copeland$^\alpha$, where α is a rational number, $0 \leq \alpha \leq 1$. These voting rules are formally defined as follows. Fix an election $E = (C, V)$ where $C = \{c_1, \ldots c_m\}$ and $V = (v_1, \ldots v_n)$. and define

$$N_E(c_i, c_j) = |\{v_k \mid c_i \succ_k c_j\}|.$$

Let α be a rational number such that $0 \leq \alpha \leq 1$. Then the Copeland$^\alpha$ score of a candidate c_i, which we denote by $\mathrm{score}_E^\alpha(c_i)$, is defined as

$$\mathrm{score}_E^\alpha(c_i) = |\{c_j \mid N_E(c_i, c_j) > N_E(c_j, c_i)\}| + \alpha|\{c_j \mid N_E(c_i, c_j) = N_E(c_j, c_i)\}|.$$

The maximin score of a candidate c_i, which we denote by $\mathrm{score}_E^m(c_i)$, is defined as $\mathrm{score}_E^m(c_i) = \min_{i \neq j} N_E(c_i, c_j)$.

Theorem 9. *Shift bribery is* NP*-complete for maximin and, for each rational* α *between* 0 *and* 1*, for Copeland*$^\alpha$.

It is interesting to compare the results of this section with those of [11], where it is shown that for irrational voters microbribery for Copeland^0 and for Copeland^1 is in P. In fact, we can show that microbribery for the case of irrational voters is in P also for Borda and maximin (though we omit these results due to limited space and our focus on rational voters). This is a further (meta)-argument that perhaps the main source of hardness in many voting problems stems from having to deal with preference orders rather than the properties of particular voting rules.

6 Conclusions

We introduced two notions of nonuniform bribery—swap bribery and shift bribery—for the standard model of elections, and analyzed their complexity for several well-known voting rules such as plurality, k-approval, Borda, Copeland, and maximin. It turns out that, in sharp contrast to the easiness results for microbribery [11] and nonuniform bribery in utility-based systems [9], swap bribery is NP-hard for many of these rules. This is quite surprising as our swap bribery is essentially the microbribery model adapted to the rational-voter setting.

Our work leads to several open problems. First, it would be useful to identify natural special cases of our setting for which one can find an optimal swap bribery in polynomial time. Another way to tackle computational hardness is by constructing efficient approximation algorithms for swap bribery and shift bribery; Theorem 8 makes the first step in this direction. Designing approximation algorithms for shift bribery under other voting rules as well as for swap bribery is an interesting topic for future research.

Acknowledgements. We would like to thank the anonymous referees for very helpful comments and suggestions. We are also grateful to Jörg Rothe for hosting a visit of the first two authors to Heinrich-Heine-Universität, Düsseldorf. Piotr Faliszewski is supported by AGH University of Science and Technology grant no. 11.11.120.777. Edith Elkind is supported by EPSRC grant GR/T10664/01, ESRC grant ES/F035845/1, and by NRF Research Fellowship.

References

[1] J. Bartholdi, III, C. Tovey, and M. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3):227–241, 1989.

[2] N. Betzler and B. Dorn. Towards a dichotomy of finding possible winners in elections based on scoring rules. In *Proceedings of MFCS-09*, 2009.

[3] S. Brams and R. Sanver. Critical strategies under approval voting: Who gets ruled in and ruled out. *Electoral Studies*, 25(2):287–305, 2006.

[4] I. Caragiannis, J. Covey, M. Feldman, C. Homan, C. Kaklamanis, N. Karanikolas, A. Procaccia, and J. Rosenschein. On the approximability of Dodgson and Young elections. In *Proceedings of SODA-09*, 2009

[5] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3), 2007.

[6] E. Elkind, P. Faliszewski, and A. Slinko. On distance rationalizability of some voting rules. In *Proceedings of TARK-09*, 2009.

[7] E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. Technical Report arXiv:0905.3885 [cs.GT], arXiv.org, May 2009.

[8] G. Erdélyi, M. Nowak, and J. Rothe. Sincere-strategy preference-based approval voting broadly resists control. In *Proceedings of MFCS-08*, 2008.

[9] P. Faliszewski. Nonuniform bribery (short paper). In *Proceedings of AAMAS-08*, 2008.

[10] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. The complexity of bribery in elections. In *Proceedings of AAAI-06*, 2006.

[11] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, 2009.

[12] M. Fellows, F. Rosamond, and A. Slinko. Sensing God's will is fixed parameter tractable. Technical Report N.561, Department of Mathematics. The University of Auckland, July 2008.

[13] E. Friedgut, G. Kalai, and N. Nisan. Elections can be manipulated often. In *Proceedings of FOCS-08*, 2008.

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

[15] K. Konczak and J. Lang. Voting procedures with incomplete preferences. In *Proceedins of the Multidisciplinary IJCAI-05 Worshop on Advances in Preference Handling*, pages 124–129, July/August 2005.

[16] T. Meskanen and H. Nurmi. Closeness counts in social choice. In M. Braham and F. Steffen, editors, *Power, Freedom, and Voting*. Springer-Verlag, 2008.

[17] T. Walsh. Uncertainty in preference elicitation and aggregation. In *Proceedings of AAAI-07*, 2007.

[18] L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. In *Proceedings of AAAI-08*, 2008.

[19] L. Xia and V. Conitzer. Generalized scoring rules and the frequency of coalitional manipulability. In *Proceedings of EC-08*, 2008.

[20] L. Xia and V. Conitzer. A sufficient condition for voting rules to be frequently manipulable. In *Proceedings of EC-08*, 2008.

[21] M. Zuckerman, A. Procaccia, and J. Rosenschein. Algorithms for the coalitional manipulation problem. *Artificial Intelligence*, 173(2):392–412, 2008.