Approximability of Manipulating Elections[1]

Eric Brelsford, Piotr Faliszewski, Edith Hemaspaandra,
Henning Schnoor and Ilka Schnoor

Abstract

In this paper, we set up a framework to study approximation of manipulation, control, and bribery in elections. We show existence of approximation algorithms (even fully polynomial-time approximation schemes) as well as obtain inapproximability results. In particular, we show that a large subclass of scoring protocols admits fully polynomial-time approximation schemes for the coalitional weighted manipulation problem and that if certain families of scoring protocols (e.g., veto) admitted such approximation schemes then P = NP. We also show that bribery for Borda count is NP-complete and that there is no approximation algorithm that achieves even a polynomial approximation ratio for bribery in Borda count for the case where voters have prices.

1 Introduction

Elections are an essential mechanism that each democratic society uses to make joint decisions. They are also important tools within computer science. For example, [DKNS01] show how to build a meta search-engine via conducting elections between other search engines; Ephrati and Rosenschein [ER97] use voting to solve certain planning problems; and in the context of multiagent systems, elections and voting are naturally used to obtain the joint decisions of agent societies.

Unfortunately, a famous result of Gibbard [Gib73] and Satterthwaite [Sat75] states that (see also the work of Duggan and Schwartz [DS00]) for any reasonable election system (with at least 3 candidates) there exist scenarios where at least some voters have an incentive to vote strategically, i.e., to vote not according to their true preferences but in a way that yields a result more desirable for them. Similarly, the result of an election can be skewed by an external agent who bribes some of the voters to change their votes or even by the authority organizing the election, via, e.g., encouraging or discouraging particular candidates from participating, or via arranging voting districts in a certain way.

The possibility that the result of the election can be skewed via strategic voting, bribery, or procedural control is very disconcerting. In the early 90s, Bartholdi, Orlin, Tovey, and Trick [BTT89, BO91, BTT92] suggested a brilliant way to circumvent this issue. They observed that since all voters are computationally-bounded entities, even if various forms of manipulating elections are possible in principle,[2] they constitute a real threat only if it is computationally easy, for a given election system, to determine the appropriate actions that affect the result in the desired way (i.e., for the case of strategic voting to determine how the manipulators should vote; for the case of bribery determine who to bribe and how, etc.). To measure the computational difficulty of manipulation and control, Bartholdi, Orlin, Tovey, and Trick used the complexity-theoretic notion of NP-hardness.

The ideas of Bartholdi, Orlin, Tovey and Trick did not receive that much attention until a few years ago, when it became apparent that elections and voting are important tools in the context of multiagent systems, and that software agents are capable of much more systematic attempts at manipulating elections than, say, humans. Thus, in recent

[1] This is an extended version of the AAAI-08 paper with the same title.

[2] In the literature the technical term *manipulation* means strategic voting. In this section by *manipulating* we mean the general notion of affecting the result of an election.

years many papers focused on the computational analysis of voting rules with respect to manipulation, e.g., [CSL07, EL05, PR07, PRZ07, HH07], bribery [FHH06, FHHR07, Fal08], and control [HHR07, FHHR07, PRZ07]. Most of these papers focus on obtaining polynomial-time algorithms and NP-hardness results for various forms of affecting the result of elections. However, NP-hardness gives only worst-case complexity guarantees, and it might very well be the case that even though, say, manipulation in a given election system is NP-hard, finding effective manipulations is often easy. Recently, Conitzer and Sandholm [CS06], and Procaccia and Rosenschein [PR07] looked into these issues and they provide examples of voting rules and distributions of votes for which this is the case. We will refer to the approach presented, among others, in these two papers as *frequency of hardness approach*.

In this paper we refine the study of the complexity of manipulating elections via analysis of approximability of NP-hard election-manipulation problems. An important contribution of this paper is a natural, uniform objective function that can be used to measure the effectiveness of a particular manipulation, bribery, or control attempt. Thus we set up a general framework for studying approximation for these problems.[3] Our function is particularly interesting for the case of manipulation, where defining a natural objective function is not straightforward.

We show existence of approximation algorithms (even fully polynomial-time approximation schemes) for manipulation for a large subclass of voting rules known as scoring protocols (for bounded numbers of candidates) as well as obtain inapproximability results regarding several prominent families of scoring protocols (e.g., veto and k-approval for unbounded number of candidates). We prove NP-hardness of bribery for Borda count and inapproximability of bribery for Borda count for the case where each voter has a price for changing its vote. Hardness results for control (i.e., changing the outcome of the election by *adding* voters) of unweighted Borda elections have been obtained by Russell [Rus07]. We use a similar technique to prove that the bribery problem for unweighted Borda is NP-complete. To the best of our knowledge, our NP-hardness result for Borda is the first hardness result for a problem of affecting the result of unweighted Borda count elections via *modifying* the voters.

Related work Several previous papers studied approximation of various manipulation and bribery problems but each of them used objective functions specifically tailored to their tasks. In particular, Faliszewski [Fal08] studied approximability of the total cost of a bribery for plurality and approval voting. Zuckerman, Procaccia and Rosenschein [ZPR08], among other things, studied approximability of the minimum number of unweighted manipulators needed to change the result of an election for several voting systems, including Borda count.

We contrast our approach and results with the frequency of hardness approach. The existence of an approximation algorithm (in particular, the existence of a fully polynomial-time approximation scheme) for a given election problem is much stronger evidence that this problem is practically easy than a frequency of hardness result stating that the problem is easy often, according to some distribution. The reason for this is that a polynomial-time approximation algorithm guarantees to find a near-optimal answer for *every* input instance. If our problem is frequently easy it might still be the case that the instances that we encounter in practice happen to be the "rare" difficult ones. On the other hand, inapproximability is a worst case notion. If a problem is in general inapproximable, it might still be the case that most of its instances are easy (are easily approximable). Nonetheless, inapproximability of a given NP-hard election problem is stronger evidence of its computational hardness than NP-hardness alone.

We focus on manipulation and bribery rather than on control, but we mention that Brelsford [Bre07] studied several control problems from the point of view of approximation.

[3]To be technically correct, our approach is limited to voting rules that assign numerical scores to the candidates. This is the case for most standard voting rules.

2 Preliminaries

Elections An election E is a pair (C, V), where C is a finite set of candidates and V a finite multiset of strict linear orders on C. An order $v \in V$ is called a vote and represents the preference of a voter over the candidates. The winner of an election E depends on the underlying election system. In this paper we consider only election systems that are represented by scoring protocols and families of scoring protocols. A scoring protocol is a vector $(\alpha_1, \ldots, \alpha_m)$ of natural numbers with $\alpha_1 \geq \cdots \geq \alpha_m$. Using this protocol the winner of an election E with m candidates can be determined as follows: Every candidate c gets α_i points for every vote that ranks c in the ith place and $\text{score}_E(c)$ is the sum of all points c gets in this way. In the end c is a winner if no other candidate has a higher score. We also allow the votes in E to have weights, in this case each vote with weight $w \in \mathbb{N}$ is counted as w identical votes.

Let $(S_i)_{i\geq 1}$ be a family of scoring protocols such that S_i is a scoring protocol of length i. We represent by $(S_i)_{i\geq 1}$ the election system that uses S_m to determine the winner of an election with m candidates. Borda count is the election system using $((i-1, i-2, \ldots, 0))_{i\geq 1}$, and veto is the election system using $((1, 1, \ldots, 1, 0))_{i\geq 1}$.

Approximating Elections In this paper we study approximation algorithms for manipulation and bribery. In both problems our goal is to ensure that a specified candidate is a winner but in manipulation we attempt to reach this goal via, in essence, adding a certain number of votes, whereas in bribery we do so via changing up to a given number of votes. (Note that sometimes manipulation is defined as allowing to *change* a *specified* set of voters. Our version allows to state our results in an easier notation—it is easy to see that these notions describe the exact same issue. One may view adding the manipulators' votes as a process where the manipulators make up their minds as to how to vote, and then cast their votes. Note that unlike for e.g., bribery, the original votes of the manipulators are completely irrelevant for the problem to determine if a designated candidate can be made a winner.) We also study the manipulation problem where the voters additionally have weights, and the bribery problem where the votes have prices which the briber has to pay in order to change the vote.

We require our approximation algorithms to produce "solutions" to their respective instances. A solution is a strategy specifying which actions to perform, i.e., what votes to add for the case of manipulation and which votes to change (and how to change them) for the case of bribery.

In our model we assume that we know all the votes that are supposed to be cast in the election. In reality, however, we are often limited to only having a guess regarding these votes. Thus we are interested in finding a strategy that benefits the specified candidate as much as possible so that this candidate has a good change of becoming a winner even if the guess is a little off.

In the setting of scoring protocols (or any other score-based election system), a natural way to measure the performance of a candidate p in an election E (written as $\text{perf}^E(p)$) is $\text{score}_E(p) - \max\{\text{score}_E(c) \mid c \in C \setminus \{p\}\}$, the difference between the score of p and that of p's strongest competitor. $\text{perf}^E(p)$ tells us "how much" p wins the election or "how close" p is to winning it. Obviously, p wins the election E if and only if $\text{perf}^E(p)$ is nonnegative.

A natural measure of the effectiveness of a manipulating action s within election E is the increase of performance of the favorite candidate p obtained by applying this action.

Definition 1 *$\beta(E, s) = \text{perf}^{s(E)}(p) - \text{perf}^E(p)$, where $s(E)$ denotes the election resulting from applying action s to E.*

Note that the β function allows us to deal with uncertainty in a natural way: If we only have knowledge about a part of the election, then it is a natural goal to give our candidate as

much of a headstart in the part of the election that we do know as possible. This is exactly what is expressed in the β-function. Also, β can be applied not only to manpulation and bribery as studied in this paper, but to just about every possible way to interfere with the result of an election. We therefore believe it to be a uniform way to describe the "success" of the action of a dishonest party in an election scheme.

Finally observe that for given strategies (added voters, bribes, etc.), the value of β can be negative. However, for scoring protocols (and most other natural election rules) it is easy to come up with strategies that have a nonnegative value of β (strategies that do nothing at all suffice). Hence we require our approximation algorithms to only output "reasonable" strategies, i.e., strategies for which the value of β is nonnegative. We now define our optimization problems (which we will prefix with the election system under consideration):

\$-bribery-max The input I consists of an election E, for each existing vote a natural number defining its price, a preferred candidate p, and a natural number k representing the budget available to the briber. Solutions consist of a set of votes in the election E such that the sum of their prices does not exceed the budget k, and new votes to replace them with. The goal is to find a solution s maximizing $\beta(E, s)$.

weighted-manipulation-max Here, the input I consists of an election E where each vote is accompanied by its weight, the preferred candidate p, and a list of weights (of the manipulators). A solution consists of a vote for each manipulator. Again, the goal is to find a solution s such that $\beta(E, s)$ is maximal.

Note that if we could compute the maximum value of β for a given optimization problem then, naturally, we could solve the corresponding decision problem. This means that if the corresponding decision problem is NP-hard (as is often the case for manipulation and bribery) then we cannot hope to compute the optimal value of β exactly. However, since β is defined as an increase in p's performance and thus its maximum value is positive in most settings, we can attempt to use natural techniques to compute it approximately.

Approximation Algorithms and Elections The quality of an approximation algorithm is usually measured by comparing the solutions it computes to the optimal ones. For our optimization problems, an instance I contains the election itself, the preferred candidate, and additional parameters limiting the possible strategies for affecting the result of the election (i.e., the available budget in \$-bribery and the weights of the manipulators in manipulation). For such an instance I containing the election E, we define an *optimal solution* to be a solution s that achieves the maximal possible value of $\beta(E, s)$ among *all* legal solutions. We define $\mathrm{OPT}(I)$ as $\beta(E, s)$ for such an optimal solution s.

Given an instance I, an approximation algorithm $\mathcal{A}$ is required to produce a legal solution $\mathcal{A}(I)$, that is, a solution that respects the constraints specified in I. For a positive real constant c, we say that $\mathcal{A}$ is a *factor c approximation algorithm*, if for each instance I containing the election E, we have that $\beta(E, \mathcal{A}(I)) \geq \frac{1}{c}\mathrm{OPT}(I)$. Such an algorithm guarantees that the effectiveness of the solution obtained from applying $\mathcal{A}$ to the instance is at least $\frac{1}{c}$ of the effectiveness that the optimal solution achieves.

We say that an algorithm $\mathcal{A}$ is a *polynomial-time approximation scheme* if for each input (I, ε), where ε is a rational value between 0 and 1 and where I contains an election E, it holds that: (a) $\mathcal{A}$ produces a solution $s = \mathcal{A}(I, \varepsilon)$ such that $\beta(E, s) \geq (1-\varepsilon)\cdot\mathrm{OPT}(I)$, and (b) for each fixed value of ε, $\mathcal{A}$ runs in time polynomial in $|I|$. If, in fact, $\mathcal{A}$ runs in time polynomial in both $|I|$ and $\frac{1}{\varepsilon}$ then $\mathcal{A}$ is a *fully polynomial-time approximation scheme* (FPTAS). In the current paper we only consider maximization problems, analogous definitions can be given for minimization problems as well.

3 Manipulation in Scoring Protocols

Hemaspaandra and Hemaspaandra [HH07] showed that for each scoring protocol $\alpha = (\alpha_1, \ldots, \alpha_m)$ such that it is not the case that $\alpha_2 = \cdots = \alpha_m$ the problem α-weighted-manipulation is NP-complete (see also [CSL07, PR07]). In this section we show that, nonetheless, weighted manipulation is easy for a large class of scoring protocols in practice. We do so via showing FPTASes for the scoring protocols in this class.

Let α be a scoring protocol. An instance I of α-weighted-manipulation-max is a tuple (E, w, p) where $E = (C, V)$ is an election with candidate set C and weighted nonmanipulative voters V, $w = (w_1, \ldots, w_n)$ is a sequence of weights of the manipulative voters, and $p \in C$ is our preferred candidate. Our goal is to maximize the performance of p. That is, our goal is to find a solution *sol* such that $\beta(E, sol) = \mathrm{OPT}(I)$.

Theorem 2 *Let $\alpha = (\alpha_0, \ldots, \alpha_m)$ be a scoring protocol such that $\alpha_0 > \alpha_1$. There is an algorithm $\mathcal{A}$ that given a rational number ε, $0 < \varepsilon < 1$, and an instance $I = (E, w, p)$ of α-weighted-manipulation-max computes, in polynomial time in $|I|$ and $\frac{1}{\varepsilon}$, a solution sol such that $\beta(E, sol) \geq (1 - \varepsilon)\mathrm{OPT}(I)$.*

Note that Theorem 2 claims that for each *separate* scoring protocol $(\alpha_0, \ldots, \alpha_m)$, where $\alpha_0 > \alpha_1$, there is a *separate* algorithm. In particular, each of the algorithms from Theorem 2 is tailored for a fixed number of candidates. Before we jump to the proof, we need to introduce some notation.

Let $\alpha = (\alpha_0, \ldots, \alpha_m)$ be a scoring protocol where $\alpha_0 > \alpha_1$ and let $C = \{p, c_1, \ldots, c_m\}$ be a set of candidates. p is our preferred candidate whose performance we want to maximize. We implicitly assume that we have a set V of nonmanipulative voters, however in this discussion the only incarnation of the nonmanipulative voters is through the sequence s below.

We let $w = (w_1, \ldots, w_n)$ be the sequence of weights of the manipulators. Naturally, to maximize p's performance, each manipulator ranks p first. The complexity of α-weighted-manipulation-max comes from the difficulty in arranging the remainder of the manipulators' votes in such a way as to minimize the score of p's most dangerous competitor.

By $\mathcal{E}(C, w)$ we mean the set of all elections over the candidate set C with voter set containing exactly voters with weights $w_1, \ldots, w_n$. Let $s = (s_1, \ldots, s_m)$ be a sequence of nonnegative integers. Intuitively, the sequence s gives the scores that candidates c_1 through c_m receive from the nonmanipulative voters. By $S_\alpha(E, s)$ we mean $\max_{i \in \{1,\ldots,m\}}\{\mathrm{score}_E(c_i) + s_i\}$ and by $T_\alpha(w, s)$ we mean $\min_{E \in \mathcal{E}(C,w)} S_\alpha(E, s)$. Function $T_\alpha(w, s)$ measures the smallest possible top score of a candidate from $\{c_1, \ldots, c_m\}$ after the manipulators cast their votes. We now prove that for each scoring protocol α there is an FPTAS for T_α.

Lemma 3 *Let $\alpha = (\alpha_0, \ldots, \alpha_m)$ be a scoring protocol and let $C = \{p, c_1, \ldots, c_m\}$. There is an algorithm $\mathcal{T}$ that given a rational number ε, $0 < \varepsilon < 1$, a sequence $s = (s_1, \ldots, s_m)$ of nonnegative integers and a sequence of manipulators weights $w = (w_1, \ldots, w_n)$ computes an election $E \in \mathcal{E}(C, w)$ such that $S_\alpha(E, s) \leq (1 + \varepsilon)T_\alpha(w, s)$. Algorithm $\mathcal{T}$ runs in polynomial time in n, m, and $\frac{1}{\varepsilon}$.*

Proof. Set $w_{\max} = \max\{w_1, \ldots, w_n\}$ and set $K = \frac{\varepsilon w_{\max}}{n\alpha_1}$. Set $w' = (K\lceil\frac{w_1}{K}\rceil, \ldots, K\lceil\frac{w_n}{K}\rceil)$. It is possible to compute in polynomial time in n, m, and $\frac{1}{\varepsilon}$ an election $E' \in \mathcal{E}(C, w')$ such that $S_\alpha(E', s) = T_\alpha(w', s)$. (One can do so via a routine dynamic programming approach; we enforce that in our solution each voter ranks p first.) Let E be an election identical to E' only that appropriate voters have weights $w_1, \ldots, w_n$ instead of $w'_1, \ldots, w'_n$. Our algorithm outputs E.

It is easy to see that our algorithm can be made to work in polynomial time as required. Let us now show that the solution it produces satisfies the requirements regarding quality.

It is easy to see that $T_\alpha(w,s) \geq \alpha_1 w_{\max}$ and that $S_\alpha(E',s) \leq T_\alpha(w,s) + \alpha_1 nK$. The former is true because some candidate needs to get α_1 points from the manipulator with weight $w_{\max}$ and the second follows from the fact that for each i in $\{1,\ldots,n\}$ we have $w_i \leq w'_i < w_i + K$. For the same reason $S_\alpha(E,s) \leq S_\alpha(E',s)$.

Thus, $S_\alpha(E,s) \leq T_\alpha(w,s) + \alpha_1 nK = T_\alpha(w,s) + \varepsilon w_{\max}$. Since $T_\alpha(w,s) \geq \alpha_1 w_{\max}$, this yields that $S_\alpha(E,s) \leq (1+\varepsilon)T_\alpha(w,s)$. (Note that, technically, this argument is only correct if $\alpha_1 \geq 1$ but, naturally, if $\alpha_1 = 0$ then the theorem is trivially satisfied.) This completes the proof. □

With Lemma 3 at hand we can prove Theorem 2.

Proof. Our input is $I = (E,w,p)$, where $E = (C,V)$ is an election with candidate set $C = \{p, c_1 \ldots, c_m\}$ and set V of nonmanipulative voters, $w = (w_1,\ldots,w_n)$ is a sequence of manipulators' weights, and p is our preferred candidate. Our goal is to find a solution *sol* (a collection of votes for the manipulators to cast) that maximizes $\beta(E, sol)$.

Let $W = \sum_{i=1}^{n} w_i$ and let $w_{\max} = \max\{w_1,\ldots,w_n\}$. For each i in $\{1,\ldots,m\}$ let $s_i = \mathrm{score}_E(c_i)$. We assume that the candidates $c_1,\ldots,c_m$ are listed in such an order that $s_1 \geq s_2 \geq \cdots \geq s_m$. Since $\alpha_0 > \alpha_1$, in every optimal solution each manipulator ranks p first and so $\mathrm{OPT}(I) = W\alpha_0 - (T_\alpha(w,s) - s_1)$. It would seem that computing approximately $T_\alpha(w,s)$ should be enough to get a good approximation of $\mathrm{OPT}(I)$, but unfortunately $T_\alpha(w,s)$ can be much bigger than $\mathrm{OPT}(I)$. We have to, in some sense, reduce its value first.

Note that we can disregard all candidates c_j such that $s_1 - s_j > \alpha_1 W$. If there are k such candidates then the manipulators may simply rank them on the first k positions after p. For the sake of simplicity, we assume that there are no such candidates.

Let $s' = (s_1 - s_m, \ldots, s_m - s_m)$. It is easy to see that $\mathrm{OPT}(I) = W\alpha_0 - (T_\alpha(w,s) - s_1) = W\alpha_0 - (T_\alpha(w,s') - s'_1)$. Additionally, via the above paragraph, we have that for each s'_i it holds that $s'_i \leq \alpha_1 W$. However, this means that $T_\alpha(w,s') \leq 2\alpha_1 W$. This is so because at worst the candidate whose score is the value of $T_\alpha(w,s')$ gets $\alpha_1 W$ points from s' and another $\alpha_1 W$ points from the manipulators.

Using algorithm $\mathcal{T}$ from Lemma 3, we fill-in the manipulators' votes to form an election $E' \in \mathcal{E}(C,w)$ such that all voters in E' rank p first and $T_\alpha(w,s') \leq S_\alpha(s',E') \leq (1+\varepsilon')T_\alpha(w,s')$, where $\varepsilon' = \frac{1}{2\alpha_1}\varepsilon$. (Recall that in our setting α_1 is a constant.) Votes obtained in this way are the solution *sol* that our algorithm produces and we have $\beta(E, sol) = W\alpha_0 - (S_\alpha(s',E') - s'_1)$. Note that

$$\begin{aligned}
\mathrm{OPT}(I) &= W\alpha_0 - (T_\alpha(w,s') - s'_1) \\
&\geq W\alpha_0 - (S_\alpha(s',E') - s'_1) \\
&\geq W\alpha_0 - ((1+\varepsilon')T_\alpha(w,s') - s'_1) \\
&= W\alpha_0 - (T_\alpha(w,s') - s'_1) - \varepsilon' T_\alpha(w,s') \\
&= \mathrm{OPT}(I) - \varepsilon' T_\alpha(w,s').
\end{aligned}$$

Since $\mathrm{OPT}(I) \geq W$ (this is a consequence of the fact that $\alpha_0 > \alpha_1$), $T_\alpha(w,s') \leq 2\alpha_1 W$, and $\varepsilon' = \frac{1}{2\alpha_1}\varepsilon$, via the above calculations, $\mathrm{OPT}(I) \geq W\alpha_0 - (S_\alpha(s',E') - s'_1) \geq (1-\varepsilon)\mathrm{OPT}(I)$ and thus $\mathrm{OPT}(I) \geq \beta(E, sol) \geq (1-\varepsilon)\mathrm{OPT}(I)$. This completes the proof. □

Interestingly, we can use Theorem 2 to obtain results similar to those of Zuckerman, Procaccia, and Rosenschein [ZPR08], but for the case of scoring protocols $\alpha = (\alpha_0,\ldots,\alpha_m)$ such that $\alpha_0 > \alpha_1$. Note that Theorem 4 below says that there is a *separate* algorithm for *each* scoring protocol of the given form. (Also, keep in mind that each *single* scoring protocol only works with a fixed number of candidates.)

Theorem 4 *Let ε be a rational number, $0 < \varepsilon < 1$, and let $\alpha = (\alpha_0, \ldots, \alpha_m)$ be a scoring protocol such that $\alpha_0 > \alpha_1$. There is an algorithm that given an instance $I = (E, w, p)$ of α-weighted-manipulation, where $w = (w_1, \ldots, w_n)$ is the sequence of manipulators' weights, has the property that if there is a successful manipulation for instance I, it finds, in polynomial time in $|I|$ and $\frac{1}{\varepsilon}$, a successful manipulation for instance $I' = (E, (w_1, \ldots, w_n, w_{n+1}), p)$, where $w_{n+1} = \lceil \varepsilon \max\{w_1, \ldots, w_n\} \rceil$.*

We omit the easy proof. (The idea is to simply find a good enough approximation and then add a single voter, with appropriate weight, that ranks p first.)

Theorem 2 notwithstanding, we now show that for the case of an unbounded number of candidates there are no FPTASes for veto-weighted-manipulation-max and for k-approval-weighted-manipulation-max, unless P $\neq$ NP.

Theorem 5 *If* P $\neq$ NP*, there is no FPTAS for veto-weighted-manipulation-max.*

To prove Theorem 5 it suffices to show that the unary version of weighted manipulation in veto, i.e., one where each weight is encoded in unary, is NP-complete. In unary-encoded variant of weighted manipulation (in veto and in each fixed scoring protocol) it holds that the maximum value of β function is polynomially bounded. Thus, if there was an FPTAS for veto-weighted-manipulation-max, then one could, via a good enough approximation, solve veto-weighted-manipulation exactly in polynomial time. This is a contradiction if P $\neq$ NP.

Theorem 6 *unary-veto-weighted-manipulation is NP-complete.*

Proof. We will reduce from the NP-complete problem Unary-3-Partition [GJ79]: Given a multiset A of $3m$ positive integers in unary and an integer bound B in unary such that for each $a \in A$, $B/4 < a < B/2$ and such that $\sum_{a \in A} a = mB$, does there exist a partition of A into m subsets $A_1, \ldots, A_m$ such that $\sum_{a \in A_i} a = B$ for all i? (Note that $\|A_i\| = 3$ for all i; hence the problem's name.)

Our reduction works as follows. The election consists of one voter of weight B with preference $c_1 > c_2 > \cdots > c_m > p$ and the manipulators have weights $a_1, \ldots, a_{3m}$.

We claim that there is a successful partition of A if and only if p can be made a winner in our constructed election. First suppose that there exists a partition of A into m subsets $A_1, \ldots, A_m$ such that $\sum_{a \in A_i} a = B$. Let the manipulators corresponding to A_i veto candidate c_i. Note that every candidate c_i receives exactly B vetoes. In the resulting election, $\text{score}(p) = mB$ (p is never vetoed), and $\text{score}(c_i) = B + mB - B = mB$, and so p is a winner of the election. For the converse, suppose the manipulators vote in such a way that p is a winner of the election. Without loss of generality, we may assume that p is never vetoed, and so $\text{score}(p) = mB$. In order for p to be a winner, $\text{score}(c_i)$ can be at most mB. $\sum_{c_i} \text{score}(c_i) = m^2 B$, and so this can only happen if $\text{score}(c_i) = mB$ for all c_i. It follows that each c_i receives exactly B vetoes. Let A_i consist of the multi-set of the weights of the voters that veto c_i. Then $A_1, \ldots, A_m$ is a partition such that $\sum_{a \in A_i} a = B$ for all i. □

The same approach can be used to show NP-hardness (and thus non-existence of FPTAS unless P = NP) for unary manipulation for many other families of scoring protocols.

Theorem 7 *If* P $\neq$ NP*, there is no FPTAS for k-veto-weighted-manipulation-max, k-weighted-approval-manipulation-max, and generalized versions of k-weighted-approval-manipulation-max where, as in k-approval, voters give only points to the first k candidates, but any $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_k > 0$ is allowed.*

4 The Bribery Problem for Borda Count

In this section, we prove hardness results for the Borda count election system.

NP-hardness of the decision version We start by showing that Borda-bribery is NP-complete. In Borda-bribery we are given an election E, a distinguished candidate p, and a nonnegative integer k, and we ask if it is possible to ensure that p is a winner of E via modifying at most k votes. Note that our result regards the simplest variant of bribery where each voter has unit weight and unit price. Hardness of more involved variants (i.e., ones including prices or weights or both) follows naturally.

Our proof works via a reduction from a specifically crafted restriction of the set cover problem.

Problem:	34-XC
Input:	Set S, $\|S\| = n$, sets $T_1, \ldots, T_{\frac{3}{4}n}$, where $\|T_i\| = 4$ for each T_i and where each $s \in S$ is in exactly 3 sets T_i.
Question:	Is there a set $I \subseteq \{1, \ldots, \frac{3}{4}n\}$ such that for $i, j \in I, i \neq j$, $T_i \cap T_j = \emptyset$ and $\bigcup_{i \in I} T_i = S$?

It easily follows from the definition that each correct solution I has exactly $\frac{1}{4}n$ elements. This problem was shown to be NP-complete in [FHS08] (there phrased as a version of 1-in-3-satisfiability which is easily seen to be the same problem).

Theorem 8 *Borda-bribery is* NP*-complete.*

Proof. For a set A of candidates, writing A in a vote means A in some arbitrary, but fixed, order. $\overleftarrow{A}$ denotes the candidates of A in reverse order.

Let $S = \{s_1, \ldots, s_n\}$, $T_1, \ldots, T_{\frac{3}{4}n}$ be an instance of 34-XC. Let $k_1 = n^4$ and $k_2 = n^4 - 8n^3 - 4n^2$ (without loss of generality, assume that n is large enough for k_2 to be positive). Our candidate set C is $\{p\} \cup S \cup P_1 \cup P_2$, where P_1 and P_2 are sets of padding candidates such that $\|P_1\| = k_1$ and $\|P_2\| = k_2$. We set $P = P_1 \cup P_2$. The voter set is defined as follows. For each set T_i, we introduce a voter who votes as follows:

$$v_i = T_i > P_1 > S \setminus T_i > P_2 > p.$$

We also introduce m votes of the form $p > S > P$ and m votes of the form $\overleftarrow{S} > p > \overleftarrow{P}$. By increasing m we increase the point differences between pairs of candidates where one candidate comes from $S \cup \{p\}$ and the other from P, without at the same time affecting the point differences between pairs of candidates where both candidates come from $S \cup \{p\}$ or both come from P. In particular, we can choose m large enough such that with bribing at most $\frac{1}{4}n$ voters, the padding candidates cannot be made to win the election. We choose such a value for m, and hence the briber only has to ensure that the candidate p has at least as many points as each candidate in S in order for p to win the election. This allows us to establish a direct correspondence between bribery attempts bribing at most $\frac{1}{4}n$ voters and the 34-XC instance, by showing that a bribery is successful if and only if the bribed votes of the form v_i correspond to a set cover (and the new votes are set in a reasonable way, i.e., they rank p first, then the padding candidates, and then the candidates in S).

In the $\frac{3}{4}n$ votes v_i introduced earlier, p does not gain any points. For each candidate s_i in S, there are exactly 3 "good votes" (votes corresponding to a set T_j where $s_i \in T_j$), and $\frac{3}{4}n - 3$ "bad votes" (corresponding to sets T_j not containing s_i). In a good vote, s_i gains at least good-min $:= (\|C\| - 4)$ points (since the worst position that s_i can be voted in here is the fourth position). On the other hand, the most points that s_i can make in a good vote is good-max $:= (\|C\| - 1)$ points, this occurs if s_i is in the first position of the vote. For the bad votes, the minimum number of points that s_i can make is bad-min $:= (\|C\| - k_1 - n)$

points (the worst position that s_i can be in for these votes is the $(k_1 + n)$th spot), and the maximum gained in a bad vote is bad-max $:= (\|C\| - k_1 - 5)$ points (the best position to be voted here is the $(k_1 + 5)$th position, since each set T_i contains exactly 4 elements).

Since the briber only has to ensure that p beats the candidates in S, we only need to consider briberies of the form where the "deleted voters" are those corresponding to some set T_i, (deleting votes of this form is obviously better for increasing the performance of p than deleting one of the votes where S and p share the $n+1$ top spots) and the added voters vote p first, then the padding candidates, and then the candidates in S (if the briber wants to make p win, then obviously the bribed voters will vote p first, and since we constructed the election in such a way that the padding candidates cannot win, we can without loss of generality assume that the candidates in S are voted last). We fix such a bribery attempt, and for each element $s_i \in S$, let t_i be the number of good votes for s_i that are deleted by the briber. We show that the bribery is successful if and only if $t_i \geq 1$ for all of the s_i, due to the cardinality restrictions, this then implies that the deleted voters correspond to an exact cover in the sense of 34-XC (since we allow exactly $\frac{1}{4}n$ voters to be bribed).

In order to prove this, we need to show the following: If for an element s_i, the number t_i is at least 1, then the maximal number of points that s_i can have in the bribed election is less than the number of points for the preferred candidate p. On the other hand, if $t_i = 0$, then the minimum number of points that s_i has in the bribed election exceeds the score of the candidate p. We now prove this claim by computing these numbers.

The maximal number of points that s_i can have if $t_i \geq 1$ (obviously, it suffices to consider the case $t_i = 1$) occurs when s_i has the maximum number of possible points in its 2 remaining good votes, and the maximal number of points in its $\frac{1}{2}n - 2$ remaining bad votes. Additionally, the candidate s_i has the above-mentioned M points gained from the votes where all the padding candidates are voted behind all of the candidates in S and the candidate p, and it gains at most $\frac{1}{4}n(n-1)$ points from the additional bribed votes (if the candidate is voted in the first spot of the S-block in each of these votes). Therefore, the maximal number of points in the bribed election for $t_i = 1$ is bribed-max $:= 2 \cdot$ good-max $+$ $(\frac{1}{2}n - 2) \cdot$ bad-max $+ M + \frac{1}{4}n(n-1)$, which is the same as $\frac{3}{4}n^2 + \frac{1}{2}nk_2 - \frac{9}{4}n + 2k_1 + 8 + M$.

On the other hand, for $t_i = 0$, the minimal number of points for a candidate s_i is the following (3 good votes remaining, plus $\frac{1}{2}n - 3$ bad votes, the M points from above, and minimally 0 points from the additional bribed votes):

bribed-min $:= 3 \cdot$good-min$+(\frac{1}{2}n-3) \cdot$bad-min$+M$, and this is $\frac{1}{2}nk_2 + \frac{7}{2}n + 3k_1 - 12 + M$.

Finally, our preferred candidate has exactly p-score $:= \frac{1}{4}n(\|C\| - 1) + M$ points, which is the same as $\frac{1}{4}n^2 + \frac{1}{4}n(k_1 + k_2) + M$.

The required inequality bribed-max $<$ p-score $<$ bribed-min now simplifies to $\frac{3}{4}n^2 - \frac{9}{4}n + 8 < \frac{1}{4}n^2 + (\frac{1}{4}n - 2)k_1 - \frac{1}{4}nk_2 < k_1 + \frac{7}{2}n - 12$. Substituting the definitions of k_1 and k_2, this is equivalent to $\frac{3}{4}n^2 - \frac{9}{4}n + 8 < n^3 + \frac{1}{4}n^2 < n^4 + \frac{7}{2}n - 12$, which is clearly true for large enough n. Since we can assume that the input instance has a sufficient size, the proof is completed. □

Nonapproximability of Bribery We now show that there are no efficient approximation algorithms for $-bribery-max. The following result does not only show that there is no polynomial-time approximation algorithm for the problem that achieves an approximation rate of a constant factor, it also excludes a polynomial relationship between results that can be achieved efficiently and the optimal solution.

Theorem 9 *For every polynomial q there is no polynomial-time approximation algorithm $\mathcal{A}$ for Borda-*$-bribery*-max such that for all instances I containing the election E, $\mathcal{A}$ computes a solution s such that $q(\beta(E, s)) \geq \mathrm{OPT}(I)$, unless* P = NP.

This result is significantly stronger than just excluding constant-ratio approximation algorithms: It also shows that for no constant c there is a polynomial-time approximation algorithm $\mathcal{A}$ that guarantees to produce a solution $\mathcal{A}(I)$ for every instance I containing the election E such that $\beta(E, \mathcal{A}(I))$ is at least $(\mathrm{OPT}(I))^{1/c}$. Also, the NP-hardness proven in Theorem 8 refers to an even more restricted version of the problem (where no prices are allowed), hence it does not directly follow from the non-approximability proof.

Proof. Let q be a polynomial and assume $\mathcal{A}$ is a polynomial time approximation algorithm for \$-bribery-max, such that for all instances I containing the election E: $q(\beta(E, \mathcal{A}(I))) \geq \mathrm{OPT}(I)$. We show that we can use $\mathcal{A}$ to decide 34-XC in polynomial time. Note that the construction is similar but easier than the one in the proof of Theorem 8.

Choose $d, n_0 \in \mathbb{N}$ such that $q(k) \leq k^d$ for all $k \geq n_0$. Let $S = \{s_1, \ldots, s_n\}, T_1, \ldots, T_{\frac{3}{4}n}$ be an instance of 34-XC. Without loss of generality assume that $n \geq n_0$. Let m be a natural number such that $m > n^{2d} + \frac{3}{4}n^2 - \frac{15}{4}n + 11$ and let $C = S \cup \{p, c_1, \ldots, c_m\}$ be a set of candidates, where p is our preferred candidate. Let $V = \left\{v_1, \ldots, v_{\frac{3}{4}n}\right\}$ be a set of votes with

$$v_i = p > T_i > c_1 > \cdots > c_m > S \setminus T_i$$

for every $i \in \left\{1, \ldots, \frac{3}{4}n\right\}$, and let $W = \{w_1, w_1' \ldots, w_l, w_l'\}$ be a set of votes with

$$\begin{aligned} w_i &= p > s_1 > \cdots > s_n > c_1 > \cdots > c_m, \\ w_i' &= s_n > \cdots > s_1 > p > c_m > \cdots > c_1 \end{aligned}$$

for every $i \in \{1, \ldots, l\}$. We set the price of each vote in V to 1 and the price of each vote in W to $\frac{1}{4}n+1$. The effect of W is that it leaves the relative scores of p and the candidates in S invariant, while increasing them relatively to the scores of the padding candidates $c_1, \ldots, c_m$. We introduce enough of these votes such that for every possible bribery, the candidates in S will always have more points than the padding candidates. Clearly a polynomial number of these votes suffices. The algorithm $\mathcal{A}$ cannot change these votes, since their cost exceeds the budget $\frac{1}{4}n$.

Let E be the election with candidates C and votes $V \cup W$. We apply $\mathcal{A}$ on the instance $I = (E, \frac{1}{4}n, p)$ and show that $q(\beta(E, \mathcal{A}(I))) > n^{2d}$ if and only if $S, T_1, \ldots, T_{\frac{3}{4}n}$ is a *yes*-instance of 34-XC. This shows that we can use $\mathcal{A}$ to decide the problem 34-XC, which can only happen if P = NP.

First note that $\mathrm{score}_E(p) = \frac{3}{4}n(n+m) + l(2m+n)$ and for each $s \in S$: $3(n+m-4) + l(2m+n) \leq \mathrm{score}_E(s) \leq 3(n+m-1) + (\frac{3}{4}n-3)(n-5) + l(2m+n)$.

Now let $S, T_1, \ldots, T_{\frac{3}{4}n}$ be a *yes*-instance of 34-XC and let $J \subseteq \left\{1, \ldots, \frac{3}{4}n\right\}$ specify an exact cover of S. We bribe in the following way: For every $i \in J$ we replace v_i by $v_i' = p > c_1 > \cdots > c_m > S$. Let $V' = \{v_i' \mid i \in J\} \cup \left\{v_i \mid i \in \left\{1, \ldots, \frac{3}{4}n\right\} \setminus J\right\}$ be the set of votes obtained from V with this bribe and E' the election with candidates C and votes $V' \cup W$. Since J is an exact cover, for every candidate $s \in S$ we changed exactly one of the votes in V where s was in a position among the top five candidates, therefore there are two votes left in V where s is in one of the first five positions and in all other votes in V s is voted among the last n candidates. Thus $\mathrm{score}_{E'}(s) \leq 2(n+m-1) + (\frac{3}{4}n-2)(n-1) + l(2m+n)$. Note that $\mathrm{score}_{E'}(p) = \mathrm{score}_E(p)$. It follows $q(\beta(E, A(I))) \geq \mathrm{OPT}(I) \geq m - \frac{3}{4}n^2 + \frac{15}{4}n - 11 > n^{2d}$.

Assume there is no exact cover for $S, T_1, \ldots, T_{\frac{3}{4}n}$. Note that $\mathcal{A}(I)$ changes exactly $\frac{1}{4}n$ votes from V and no vote from W. Let $E_{\mathcal{A}}$ be the election obtained from E by applying the bribing strategy $\mathcal{A}(I)$. Since there is no exact cover, there is a candidate $s \in S$ such that for all $v_i \in V$ with $s \in T_i$ it holds that v_i is not changed by $\mathcal{A}(I)$ and thus v_i is a vote in $E_{\mathcal{A}}$. That means there are at least three votes in $E_{\mathcal{A}}$ that rank s among the first 5 candidates, thus we get the lower bound $\mathrm{score}_{E_{\mathcal{A}}}(s) \geq 3(n+m-4) + l(2m+n)$. By assuming that

p is ranked first in all votes it follows $\text{score}_{E_{\mathcal{A}}}(p) \leq \frac{3}{4}n(n+m-1) + l(2m+n)$. Hence $\beta(E, \mathcal{A}(I)) \leq \frac{3}{4}n^2 - \frac{27}{4}n + 24$. W.l.o.g. we can assume that n is large enough to ensure that $\frac{3}{4}n^2 - \frac{27}{4}n + 24 \leq n^2$. Then $q(E, \beta(A(I))) \leq q(n^2) \leq n^{2d}$, concluding the proof. □

The above proof also works for a variant of the bribery problem where the voters have boolean indicators whether they can be bribed or not (instead of prices).

5 Acknowledgments

Supported in part by NSF grants CCF-0426761 and IIS-0713061, a Friedrich Wilhelm Bessel Research Award, the Alexander von Humboldt Foundation's TransCoop program, and the DAAD postdoc program. We thank the anonymous AAAI and COMSOC referees for their very helpful comments.

References

[BO91] J. Bartholdi, III and J. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8(4):341–354, 1991.

[Bre07] E. Brelsford. Approximation and elections. Master's thesis, Rochester Institute of Technology, Rochester, NY, May 2007.

[BTT89] J. Bartholdi, III, C. Tovey, and M. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3):227–241, 1989.

[BTT92] J. Bartholdi, III, C. Tovey, and M. Trick. How hard is it to control an election? *Mathematical and Computer Modeling*, 16(8/9):27–40, 1992.

[CS06] V. Conitzer and T. Sandholm. Nonexistence of voting rules that are usually hard to manipulate. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 627–634. AAAI Press, July 2006.

[CSL07] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3):Article 14, 2007.

[DKNS01] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International World Wide Web Conference*, pages 613–622. ACM Press, March 2001.

[DS00] J. Duggan and T. Schwartz. Strategic manipulability without resoluteness or shared beliefs: Gibbard–Satterthwaite generalized. *Social Choice and Welfare*, 17(1):85–93, 2000.

[EL05] E. Elkind and H. Lipmaa. Hybrid voting protocols and hardness of manipulation. In *The 16th Annual International Symposium on Algorithms and Computation, ISAAC 2005*, pages 206–215. Springer-Verlag *Lecture Notes in Computer Science #3872*, December 2005.

[ER97] E. Ephrati and J. Rosenschein. A heuristic technique for multi-agent planning. *Annals of Mathematics and Artificial Intelligence*, 20(1–4):13–67, 1997.

[Fal08] P. Faliszewski. Nonuniform bribery (short paper). In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, pages 1569–1572, May 2008.

[FHH06] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. The complexity of bribery in elections. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 641–646. AAAI Press, July 2006.

[FHHR07] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Llull and Copeland voting broadly resist bribery and control. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 724–730. AAAI Press, July 2007.

[FHS08] P. Faliszewski, E. Hemaspaandra, and H. Schnoor. Copeland voting: Ties matter. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, pages 983–990, May 2008.

[Gib73] A. Gibbard. Manipulation of voting schemes. *Econometrica*, 41(4):587–601, 1973.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[HH07] E. Hemaspaandra and L. Hemaspaandra. Dichotomy for voting systems. *Journal of Computer and System Sciences*, 73(1):73–83, 2007.

[HHR07] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, 171(5-6):255–285, April 2007.

[PR07] A. Procaccia and J. Rosenschein. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research*, 28:157–181, February 2007.

[PRZ07] A. Procaccia, J. Rosenschein, and A. Zohar. Multi-winner elections: Complexity of manipulation, control, and winner-determination. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1476–1481. AAAI Press, January 2007.

[Rus07] Nathan F. Russell. Complexity of control of borda count elections. Master's thesis, Rochester Institute of Technology, July 2007.

[Sat75] M. Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2):187–217, 1975.

[ZPR08] M. Zuckerman, A. Procaccia, and J. Rosenschein. Algorithms for the coalitional manipulation problem. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 277–286, January 2008.

Eric Brelsford, Edith Hemaspaandra, Henning Schnoor, Ilka Schnoor
Department of Computer Science
Rochester Institute of Technology
Rochester, NY 14623 USA Email: `eh@cs.rit.edu`

Piotr Faliszewski
Department of Computer Science
University of Rochester
Rochester, NY 14627 USA