Journal of Artificial Intelligence Research 66 (2019) 1057–1098 Submitted 06/2019; published 12/2019

Approximating Weighted and Priced Bribery in Scoring Rules

Orgad Keller ORGAD.KELLER@GMAIL.COM
Department of Computer Science, Bar-Ilan University, Israel

Avinatan Hassidim AVINATAN@CS.BIU.AC.IL
Department of Computer Science, Bar-Ilan University, Israel

Noam Hazon NOAMH@ARIEL.AC.IL
Department of Computer Science, Ariel University, Israel

Abstract

The classic BRIBERY problem is to find a minimal subset of voters who need to change their vote to make some preferred candidate win. Its important generalizations consider voters who are weighted and also have different prices. We provide an approximate solution for these problems for a broad family of scoring rules (which includes Borda, t-approval, and Dowdall), in the following sense: for constant weights and prices, if there exists a strategy which costs Ψ, we efficiently find a strategy which costs at most $\Psi + \widetilde{O}(\sqrt{\Psi})$. An extension for non-constant weights and prices is also given.

Our algorithm is based on a randomized reduction from these BRIBERY generalizations to weighted coalitional manipulation (WCM). To solve this WCM instance, we apply the Birkhoff-von Neumann (BvN) decomposition to a fractional manipulation matrix. This allows us to limit the size of the possible ballot search space reducing it from exponential to polynomial, while still obtaining good approximation guarantees. Finding a solution in the truncated search space yields a new algorithm for WCM, which is of independent interest.

1. Introduction

In the well-studied preferential model, an election consists of a set of n voters who need to decide on a winner among a set of candidates C.[1] In order to do so each voter reveals his ballot in the form of a linear ordering of the candidates according to his preference. We also call this his *preference order*. For example, if the candidate set is $\{c_1, c_2, c_3\}$, a voter might submit the preference order $c_1 \succ c_3 \succ c_2$. By ranking—for example—c_1 before c_3, the voter indicates that he prefers c_1 to c_3. The collection of the preference orders submitted by all voters is known as the *preference profile*. Following the submission of the preference profile, the winner is determined according to some protocol, or *voting rule*.

Ideally, we would like the voters to be truthful: namely, that the preference order submitted by each voter will correspond to his true preference over the alternatives. When the voters might have the incentive to do otherwise, we refer to the voting rule as *manipulable*. Unfortunately, a celebrated result in social choice theory achieved independently by Gib-

1. Voters are also called agents, and candidates are also called alternatives, emphasizing the fact that such alternatives might not necessarily correspond to people. We will mainly use voters and candidates.

bard (1973) and Satterthwaite (1975) shows that when the number of candidates is at least 3, any reasonable voting rule is manipulable.

Manipulation can take several different forms. In Bribery—introduced by Faliszewski, Hemaspaandra, and Hemaspaandra (2009)[2] as a natural way to model the act of influencing the outcome of elections by changing the preference of a subset of the voters—an interested party preferring a specific candidate p is willing to pay some of the voters to change their vote into a given ballot, such that p will prevail. In (unweighted) *coalitional manipulation* (UCM), an interested party is willing to convince additional people to join the election as voters and vote using a strategy supplied by the party of interest. We refer to these additional voters as *manipulators*. All original voters are assumed to be truthful with known preferences. UCM models the event of canvassing, especially when the targeted audience comprises people who were not planning to exercise their right to vote. In both cases, the goal is to find a strategy that makes p win using minimal resources, for example, the number k of bribed voters or manipulators, respectively.

Several natural generalizations of the problems exist; for instance, in their *weighted* counterparts (Weighted-Bribery and WCM, respectively), all voters have associated weights; essentially this means that the ballot of a voter having weight w is counted as if it was replaced with w unweighted copies of it. Such weighted settings are common, for example, in the votes of company shareholders, and in some political scenarios (such as the European council) where voters are delegates who represent states, which are weighted, e.g., by their population size. For Bribery, in addition to the basic, unit-price variant, where all voters have a unit price for changing their ballot, other price functions exist: in \$Bribery, each voter ℓ has his own price ψ_ℓ for changing his ballot. We will consider both generalizations simultaneously (i.e., Weighted-\$Bribery). Another important scenario which is beyond the scope of the present work, is modeled by the *destructive* variants, in which the sole purpose of the manipulative party-of-interest is to prevent the currently leading candidate from winning.

The importance of the aforementioned manipulation forms stretches beyond their immediate definition. First, various manipulation forms model various aspects of *campaign management*. For instance, which electorate or demographic should a candidate target during her campaign? Where should her campaign manager direct the campaign funds? In recent years, the significance of targeting specific electorates in election campaigns has skyrocketed (see article by Peters & Alcindor, 2016, for one example of many). In this context, Bribery models the act of targeting prospective voters and convincing them to change their vote, and UCM models the act of urging people who are not planning to vote to exercise their right to do so (when this is done by an interested third party supporting a specific candidate).

Second, the problems can be formulated as optimization problems in which the goal is to find the minimum k enabling p to win. In this context, they can be treated as a measure of how far candidate p is from winning the election, or equivalently, they define the notion of a *margin* by which she loses, or the effort needed to sufficiently promote her to win (Faliszewski, Skowron, & Talmon, 2017). It can be argued that such measures are more

2. Their journal paper was preceded by a conference version from 2006.

robust compared to intrinsic measures like the candidate score in the election: for instance, they are comparable across different voting rules and election types.

In this paper, we shall focus on one of the most important families of voting rules, known as *(positional) scoring rules*, which are defined as follows. For a given *score vector* $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_{|\mathsf{C}|-1})$, where $\alpha_0 \geq \cdots \geq \alpha_{|\mathsf{C}|-1} \geq 0$, a scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ is a voting rule where each voter awards $\alpha_0, \ldots, \alpha_{|\mathsf{C}|-1}$ points to the candidates he ranked in places $0, 1, \ldots, |\mathsf{C}| - 1$, respectively. The winners are the candidates with the maximum aggregate score. A prominent example of a scoring rule is the *Borda rule*, for which $\boldsymbol{\alpha} = (|\mathsf{C}|-1, |\mathsf{C}|-2, \ldots, 0)$, i.e., every voter awards 0 points to to the candidate he ranked last, 1 to the candidate before her, 2 to the one before her, etc. Other popular cases of scoring rules are the Plurality, Veto, and t-approval voting rules.

With the Gibbard-Satterthwaite Theorem providing a pessimistic view as to the potential effect of manipulation, extensive research has focused on providing hope in the battle against manipulation by means of computational hardness. In a seminal paper by Bartholdi, Tovey, and Trick (1989), who focused on the single manipulator case, as well as in many other works which followed it, it was shown that for many voting rules, while being susceptible to manipulation, it is computationally-hard to discover the manipulation strategy itself.

Research then shifted to the coalitional variants described above. For several common voting protocols, and several forms of manipulation, it was shown that computing a successful voting strategy for the manipulators is NP-hard, see surveys by Faliszewski and Procaccia (2010), Conitzer and Walsh (2016), and Faliszewski and Rothe (2016).

In the following paragraphs we briefly describe some of the previous work on scoring rules, which places our work in context. Please refer to Section 5 for a more elaborate overview.

For Bribery, Faliszewski et al. (2009) first studied the various Bribery settings w.r.t. Plurality. They showed that Plurality-Weighted-$Bribery is NP-hard, but that both Plurality-$Bribery and Plurality-Weighted-Bribery are easy. Moving away from Plurality, the situation quickly changes: t-approval-Bribery is NP-hard (Lin, 2012) for general values of t, and Borda-Bribery is NP-hard as well (Brelsford, Faliszewski, Hemaspaandra, Schnoor, & Schnoor, 2008). More generally, for any non-Plurality-like scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, $\mathcal{R}_{\boldsymbol{\alpha}}$-Weighted-Bribery is NP-hard (Faliszewski et al., 2009) even for fixed values of $|\mathsf{C}|$. In order to overcome the hardness, several efforts were made to approximate k. Of interest is Faliszewski's (2008) fully polynomial-time approximation scheme (FPTAS) for Plurality-Weighted-$Bribery. Additional results concern cost models in which different types of bribery operations have different prices, see e.g., Elkind, Faliszewski, and Slinko (2009), Elkind and Faliszewski (2010). Further results are detailed in Faliszewski and Rothe's (2016) Bribery survey.

The equivalent landscape for UCM and WCM is richer. We provide an overview of the results in Section 5. Nonetheless, note that two main objectives for approximation appear in the literature. The first—as mentioned before—tries to approximate k, the number of manipulators (e.g., Zuckerman, Procaccia, & Rosenschein, 2009; Xia, Conitzer, & Procaccia, 2010). The second tries to approximate the minimal *score margin*—that is, the difference in points—between the highest non-preferred candidate and p, or some additive function thereof (e.g., Brelsford et al., 2008; Keller, Hassidim, & Hazon, 2019).

We aim to fill the gap in approximations for Bribery, by providing results that hold for broad families of scoring rules (encompassing well-known ones, like t-approval variants and Borda) and for various Bribery models. At the same time, we show a new connection between Bribery and WCM with the margin minimization objective, and use it for our Bribery algorithm. Specifically, we show that approximating the score margin for WCM translates to approximations on the number of manipulators (i.e., bribed voters) for Bribery. While doing so, we also advance the state-of-the-art in approximating WCM. Our methods are based on relaxed linear programs that are transformed into a valid solution (i.e., a manipulation strategy) using a seminal result from the interplay of combinatorics and geometry, namely the *Birkhoff-von Neumann (BvN) decomposition* (Birkhoff, 1946; von Neumann, 1953; Kőnig, 2001), and specifically the constructive proof to its related theorem. We use them as a tool to reduce the size of the valid strategy search space from exponential to polynomial. It thus provides an important insight about the underlying combinatorial properties of manipulation under scoring rules, and is interesting in its own right.

A recent conference publication by the authors (Keller, Hassidim, & Hazon, 2018) focused on the classic unweighted setting where the goal is to minimize the number of bribed voters. There we showed that for many scoring rules, if there exists a strategy that makes a preferred candidate p win by bribing k voters, then we can efficiently find a strategy that will make her win while bribing $\widetilde{O}(\sqrt{k})$ additional voters, with a high probability.[3]

In this work we provide similar results even for Weighted-Bribery, \$Bribery, and Weighted-\$Bribery. Specifically, if the voter weights and prices are constant in the input size, then:

> *If there exists a strategy that makes a preferred candidate p win with a cost of at most Ψ, then we can* efficiently *find a strategy that will make her win with an additional cost of $\widetilde{O}(\sqrt{\Psi})$, with high probability.*

We provide strong generalizations for the cases in which the voters' weights and prices are not constant in the input size, without adding much to the $\widetilde{O}(\sqrt{\Psi})$ factor.

It should be noted that it is a non-trivial feat to provide a guarantee tighter than any constant-factor multiplicative approximation. To provide an intuition as to why a constant-factor approximation is sometimes relatively simple, let us focus on t-approval Bribery. If k^{OPT} is the optimal number of manipulators needed, then the gap between any candidate and p can be at most $2k^{\mathsf{OPT}}$, as bribing a voter can decrease the gap between some candidate and p by at most 2. Now imagine the following strategy: iteratively, pick a voter who did not vote for p (if no such voter exists, then p is already winning), and bribe him to transfer one point from any candidate he currently supports, to p. This decreases the aforementioned gap by at least 1, and therefore at most $2k^{\mathsf{OPT}}$ bribed voters are required by this procedure. Thus, we have just described a 2-multiplicative approximation to t-approval-Bribery.

En route to providing our approximation for Weighted-\$Bribery using the reduction to min-margin-WCM, we also present new approximations for WCM for both mentioned approximation objectives. The advantage of our new method (compared to e.g., the work by Keller et al., 2019) is that when it is used as a sub-procedure of our bribery algorithm, the additive term it introduces into the overall approximation factor is dominated (in the asymptotic sense) by the additive terms incurred by other stages of the bribery algorithm.

3. The $\widetilde{O}(\cdot)$ notation suppresses factors which are poly-logarithmic in n and $|\mathsf{C}|$.

1.1 Our Results and Contributions

We assume an unbounded number of candidates $|\mathsf{C}|$. We consider two broad classes of scoring rules that we call *constant scoring rules* and *non-concentrated scoring rules*. These classes contain the well-known scoring rules, like Borda (and its truncated forms), t-approval and its specific cases of Plurality and Veto. In constant scoring rules, as the name suggests, all values in $\boldsymbol{\alpha}$ are integers which are constant in the input size. In non-concentrated scoring rules, the average score when excluding α_0, i.e., $\bar{\alpha} = 1/(|\mathsf{C}|-1) \cdot \sum_{i=1}^{|\mathsf{C}|-1} \alpha_i$ is at most $(1-\epsilon)\alpha_0$, for some constant $\epsilon > 0$.

We summarize our results in the following subsections.

1.1.1 Weighted-$Bribery

Consider Weighted-$Bribery under constant or non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, and let Ψ^{OPT} be the minimum cost required in order to make p win. Then:

- If the voter weights and prices are constant in the input size, then there exists a polynomial-time randomized algorithm finding a strategy having a cost of at most $\Psi^{\mathsf{OPT}} + \widetilde{O}(\sqrt{\Psi^{\mathsf{OPT}}})$ with high probability (Corollary 23).
- If the voter weights and prices are not constant in the input size, then there exists a similar result where the additive factor depends also on the maximum weight and price (Theorem 22).

Compared to previous results, to the best of our knowledge we are the first to provide approximations for a vast family of scoring rules, and in the classic cost settings.

1.1.2 WCM

Consider WCM under any scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$. Let T (resp. T^{OPT}) be the maximum score of a non-preferred candidate as obtained by our algorithm (resp. according to the optimal strategy). Minimizing T is equivalent to minimizing the score margin, and any additive approximation to the optimal maximum competitor score T^{OPT} is a same-factor additive approximation to the optimal score margin.

- We provide a randomized algorithm for WCM (resp. UCM) which finds a strategy that obtains a bound $T \leq T^{\mathsf{OPT}} + \widetilde{O}(\alpha_1 \sqrt{w^M_{\max} \cdot W_M})$ (resp. $T \leq T^{\mathsf{OPT}} + \widetilde{O}(\alpha_1 \sqrt{k})$) with a high probability, where $w^M_{\max}$ is the maximum manipulator weight, and W_M is the sum of manipulator weights (Theorem 5).
- Based on the previous result, we can also derive novel approximation results w.r.t. the number of manipulators. Let k^{OPT} be the minimum number of manipulators required to make p win. For non-concentrated scoring rules, we provide a randomized algorithm for UCM which, with a high probability, finds a strategy using at most $k^{\mathsf{OPT}} + \widetilde{O}(\sqrt{k^{\mathsf{OPT}}})$ manipulators (Theorem 7).

The main advantages of our methods are:

- They support many scoring rules (all scoring rules in the case of the first WCM result), compared to e.g. Zuckerman et al.'s (2009) results (for manipulation) and Faliszewski's (2008) FPTAS (for bribery).

- As detailed further in Section 5, some previous work (e.g., Brelsford et al., 2008; Faliszewski et al., 2009) had made the simplifying assumption—when providing results for general scoring rules $\mathcal{R}_{\boldsymbol{\alpha}}$—that the number of candidates $|\mathsf{C}|$ is constant. They support this view by assuming that $\boldsymbol{\alpha}$ is a fixed vector within their algorithms and thus—since it is of size $|\mathsf{C}|$—$|\mathsf{C}|$ must be seen as constant. We do not make this limiting assumption for two main reasons. First, we can always assume that our algorithms are universal in the sense that they accept $\mathcal{R}_{\boldsymbol{\alpha}}$ as part of the input. Second, the interesting scoring rules are the ones that can be succinctly encoded; for instance, Borda can be encoded by the recurrence relation $\alpha_{i+1} = \alpha_i - 1$ with the initial condition $\alpha_0 = |\mathsf{C}| - 1$.

- They generalize to the weighted setting and to voters having varied prices.

- The second of the results for coalitional manipulation, which applies to UCM, has an advantage over previous methods for the specific case of non-concentrated scoring rules. It improves the additive $(|\mathsf{C}| - 2)$-approximation given by Xia et al. (2010) when $k^{\mathsf{OPT}} = O(|\mathsf{C}|^{2-\delta})$ for some constant $\delta > 0$.

Our algorithms are worst-case and do not rely on any assumption on the distribution of voter ballots and weights: they are guaranteed to supply the guaranteed approximation factors regardless of such distributions. Specifically, and for the avoidance of any doubt, they do not rely on the votes being independent of one another, and on the weights being small.

2. Preliminaries

Here we detail some basic definitions, as well as the problem definitions, notation and prerequisites.

2.1 Basic Definitions

We first define the basic scenario of elections under positional scoring rules.

Candidate Set. Let $\mathsf{C} = \{c_0, c_1, \ldots, c_m\}$ be a candidate set consisting of the preferred candidate $p = c_0$ and the $m = |\mathsf{C}| - 1$ non-preferred candidates $c_1, \ldots, c_m$. Hereafter we refer to the non-preferred candidates as *competitors*, and let $\mathsf{C}' = \mathsf{C} \setminus \{p\}$ denote the set of competitors. As mentioned, $|\mathsf{C}'| = m$ is not assumed to be constant.

Election. An election $E = (\mathsf{C}, V)$ is defined by a candidate set C and a set $N = \{1, \ldots, n\}$ of n voters where each voter submits a *preference order*, i.e., a ranking of the candidates according to his preference. Formally, $V = (v_1, \ldots, v_n)$ is the *preference profile*, that is a list of the preference orders v_ℓ for each voter $\ell \in N$. For example, $v_\ell = c_1 \succ p \succ c_2$ is one such possible preference order if $\mathsf{C} = \{p, c_1, c_2\}$, where $\succ$ is a transitive and antisymmetic relation with the meaning that if $c \succ c'$, then c is preferred over c'. For convenience, we sometimes also treat a preference order v_ℓ as a function such that $v_\ell(c)$ is the rank of a candidate c, i.e., its (0-based) index in the preference order. For example, if $v_\ell = c_1 \succ p \succ c_2$, then $v_\ell(c_1) = 0$, $v_\ell(p) = 1$, and $v_\ell(c_2) = 2$.

Given $E = (\mathsf{C}, V)$, some *decision rule* $\mathcal{R}$ is applied in order to decide on the winner(s).

Weighted Election. A weighted election $E = (\mathsf{C}, V, \mathbf{w})$ is defined similarly to an election, with the following twist: a weight-vector $\mathbf{w}$ of dimension n of positive integers is given as part of the input as well. $\mathbf{w}$ represents the weights of the n voters, with the following meaning: the ballot v_ℓ of a voter ℓ with a weight w_ℓ is considered as if it is replaced by w_ℓ identical but unweighted copies of itself. That is, when applying the voting rule $\mathcal{R}$ to the election, we first reduce it to an unweighted election by repeating each ballot according to its respective voter's weight.

Winning Model. We assume the non-unique-winner/co-winner model where p is considered a winner even if she is not the only winner.

Positional Scoring Rules. A (positional) scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ is described by a vector $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{|\mathsf{C}|-1})$ for which $\alpha_0 \geq \alpha_1 \geq \cdots \geq \alpha_{|\mathsf{C}|-1} \geq 0$, which is used as follows: each voter awards α_i to the candidate ranked in (0-based) position i. Finally, the winning candidate is the one with the highest aggregated score. In the specific case of the Borda scoring rule, we have that $\boldsymbol{\alpha} = (|\mathsf{C}|-1, |\mathsf{C}|-2, \ldots, 1, 0)$. In t-approval, $\boldsymbol{\alpha} = (\mathbf{1}^t; \mathbf{0}^{|\mathsf{C}|-t})$ where $\mathbf{0}^{t'}$ (resp. $\mathbf{1}^{t'}$) is 0 (resp. 1) concatenated t' times. Plurality (resp. Veto) is the specific case of 1-approval (resp. $(|\mathsf{C}|-1)$-approval). In the Dowdall rule—also known as the harmonic scoring rule—$\boldsymbol{\alpha} = (1, 1/2, 1/3, \ldots, 1/|\mathsf{C}|)$.

Constant and Non-Concentrated Scoring Rules. A scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ is called *constant* if the values in $\boldsymbol{\alpha}$ are integral, and $\alpha_0 = O(1)$. A scoring rule is called *non-concentrated* if $\bar{\alpha} \leq (1-\epsilon)\alpha_0$, for some constant $\epsilon > 0$, where $\bar{\alpha} = 1/(|\mathsf{C}|-1) \cdot \sum_{j=1}^{|\mathsf{C}|-1} \alpha_j = 1/m \cdot \sum_{j=1}^{m} \alpha_j$ is the average of the values in $\boldsymbol{\alpha}$ excluding α_0.

These two classes encompass the well-known scoring rules: t-approval is a constant scoring rule, where Plurality is also non-concentrated (more generally, when $t \leq (1-\epsilon)(|\mathsf{C}|-1)$ for some constant $\epsilon > 0$, then t-approval is also non-concentrated). Borda is non-concentrated as $\bar{\alpha} = (|\mathsf{C}|-1)/2 = (1-1/2)\alpha_0$, and Dowdall is non-concentrated as $\bar{\alpha} \leq \ln(|\mathsf{C}|)/(|\mathsf{C}|-1) \ll 1 = \alpha_0$. Another example, mentioned by Put and Faliszewski (2016), is exponential-Borda, defined by $\boldsymbol{\alpha} = (2^{|\mathsf{C}|-1}, 2^{|\mathsf{C}|-2}, \ldots, 2^1, 2^0)$. For exponential-Borda, it holds that $\bar{\alpha} = (2^{|\mathsf{C}|-1}-1)/(|\mathsf{C}|-1) = o(\alpha_0)$, thus it is non-concentrated as well.

For a somewhat artificial example of a scoring rule which is neither constant nor non-concentrated, consider $\boldsymbol{\alpha} = (|\mathsf{C}|^2, |\mathsf{C}|^2-1, |\mathsf{C}|^2-2, \ldots, |\mathsf{C}|^2-|\mathsf{C}|+1)$ and notice that $\bar{\alpha} > \alpha_{|\mathsf{C}|-1} > |\mathsf{C}|^2 - |\mathsf{C}| = (1-1/|\mathsf{C}|)\alpha_0$.

2.2 Problem Definitions

We define the following problems of interest, in their simpler, unweighted version.

SR-$Bribery. Given an election E under a scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, a price ψ_ℓ for each voter ℓ, and a preferred candidate p, the goal is to make p win, by bribing a subset of voters having a minimum overall price Ψ. Prices are assumed to be positive integers. Bribing a voter is the act of replacing his ballot by a preference order to our choosing. The output is thus the identity of the bribed voters along with their new ballots.

Min-manipulator-SR-UCM. Given an election E under a scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ and a preferred candidate p, the goal is to add the least amount of additional voters (manipulators), and to determine their strategies, such that p will win.

Min-margin-SR-UCM. Given an election E under a scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, a preferred candidate p, and the number of allowed manipulators k, the goal is to determine the manipulator strategies, such that the margin $\max_{c \in \mathsf{C}'} s(c) - s(p)$ is minimized, where $s(c')$ is c''s final score. Notice that as the number of manipulators is limited, p will not necessarily win.

Note the following remarks:

- Prepending "SR-" to the basic names of the problems above is to indicate that they accept various scoring rules $\mathcal{R}_{\boldsymbol{\alpha}}$, and thus $\boldsymbol{\alpha}$ is regarded as part of the input. As such, the algorithms we will provide will be universal in the sense that they are not limited to one specific scoring rule, and that the representation of such a specific scoring rule is not hard-coded into them. In contrast, when discussing algorithms from the literature which operate on a specific rule $\mathcal{R}$ (i.e., $\mathcal{R}$ is hard-coded, either succinctly or not, in the algorithm), we will prepend either the rule name or the variable $\mathcal{R}$ to the problem name. For instance, $\mathcal{R}$-Bribery or Borda-$Bribery.

- Notice the similarity in the definitions of coalitional manipulation and bribery. In both scenarios, we need to decide on a strategy for a set of manipulators M. However, in the manipulation problems it holds that $M \cap N = \emptyset$, i.e., the manipulators are *additional* voters (besides the original ones). In contrast, for the bribery problems, $M \subseteq N$ and therefore an additional aspect of the problem is selecting voters for inclusion in M.

- Both UCM variants can be seen as optimization versions of the classic UCM decision problem, in which we are given k as part of the input, and need to answer whether k manipulators are enough in order to make p win.

For all the above problems, we consider the more general weighted variant (SR-Weighted-$Bribery, min-manipulator-SR-WCM, and min-margin-SR-WCM, respectively) where the election is a weighted election.

For the specific case of min-margin-SR-WCM, as we focus on scoring rules $\mathcal{R}_{\boldsymbol{\alpha}}$, we can define the *score profile* $\boldsymbol{\sigma}$ such that $\sigma(c)$ is the initial score of c (that is, the score of c given the non-manipulator votes). Having σ in the input makes V redundant. Notice that minimizing the margin boils down to minimizing the maximum competitor score $T = \max_{c \in \mathsf{C}'} s(c)$, as $s(p)$ is determined in advance (every manipulator will award her the maximum score possible and thus $s(p) = \sigma(p) + k\alpha_0$). Therefore, we can effectively discard p when solving the problem, and use $\boldsymbol{\alpha}' = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ instead of $\boldsymbol{\alpha}$ (i.e., α_0 is excluded). Thus min-margin-SR-WCM can be seen as a min-max problem: minimizing the maximum competitor score T.

2.3 Notation

We let $\mathcal{N} = n|\mathsf{C}|$ denote the natural size of the preference profile, i.e., the number of values it is comprised of. As a corner case, notice that if $\mathcal{N}$ is constant, then the problems become easy as both the number of voters and the number of candidates are constant. Let $[m] = \{1, \ldots, m\}$. For two parameters a, b, we denote the continuous set $[a - b, a + b]$ as $[a \pm b]$.

When a voter ℓ having weight w_ℓ ranks some candidate c in (0-based) position j, the following statements are equivalent: ℓ awards a score α_j to c (or c was awarded a score of α_j by ℓ), and c is given $\alpha_j w_\ell$ points by ℓ. In words, when discussing 'points' we already factor in the effect of the weight of the voter. On the other hand, when discussing the score of a candidate at a specific time, we refer to the number of points awarded to the candidate.

We let $w_{\max}$ be the maximum voter weight, and $\psi_{\max}$ be the maximum voter price. For any set U of voters, we let $W_U = \sum_{\ell \in U} w_\ell$ be the sum of the weights of voters in U, $\Psi_U = \sum_{\ell \in U} \psi_\ell$ be the sum of the prices of voters in U, and $w^U_{\max} = \max_{\ell \in U} w_\ell$ be the maximum weight of a voter in U.

2.4 Prerequisites

All of the new algorithms in this work are randomized. In many of the lemmas used to analyze their behavior, a constant $\lambda > 1$ will be used, and its value will be determined later in the main theorems. Such lemmas will contain mathematical expressions that are said to hold with a probability of at least $1 - c\mathcal{N}^{-\lambda+d}$ for some constants c and $d < \lambda$. That is, their *failure* probability is arbitrarily-chosen polynomially-small (in $\mathcal{N}$), based on our selection of λ, where 'failure' refers to the event that the discussed expression does not hold, and later—when discussing the main theorems—to the event that algorithm does not provide the desired approximation guarantee. Sometimes we will write "with a failure probability of at most $c\mathcal{N}^{-\lambda+d}$" when presenting such expressions, and sometimes we will informally use the term "with a high probability" when the exact probability is clear from context. As the aforementioned λ is a constant, it would not have an effect on the asymptotic behavior of the approximation factor.

We will put to heavy use the following concentration inequality, which follows quite directly from some of the well-known forms of the Chernoff bound when focusing on their asymptotic behavior.

Lemma 1. *Let $X_1, \ldots, X_n$ be independent random variables where $X_i \in [0, u]$ for all i, λ be some constant, and $\mathcal{N}$ be some large enough value. Let $X = \sum_{i=1}^{n} X_i$. Then*

$$\Pr[X \notin [\mathbb{E}[X] \pm R_1(\lambda, u, \mathbb{E}[X])]] \leq \mathcal{N}^{-\lambda} ,$$

where $R_1(\lambda, u, \mathbb{E}[X]) = 6\lambda \max\{\sqrt{u\mathbb{E}[X]}, u\} \ln \mathcal{N}$ is the deviation we allow w.r.t. the expected value $\mathbb{E}[X]$.

See further discussion and proof in the appendix. The choice of notation for the value $\mathcal{N}$ in Lemma 1 was made since in our work it will always coincide with $\mathcal{N} = n|\mathsf{C}|$, the size of the preference profile. The deviation term defined above, $R_1(\lambda, u, \mathbb{E}[X]) = 6\lambda \max\{\sqrt{u\mathbb{E}[X]}, u\} \ln \mathcal{N}$, will be used extensively in the approximation analysis of our algorithms.

One of the novel methods we shall use will be based on the seminal *Birkhoff-von Neumann (BvN) theorem* (Birkhoff, 1946; von Neumann, 1953; Kőnig, 2001), and the matrix decomposition it relies upon. This theorem studies the following families of matrices.

Doubly-Stochastic Matrices. A *doubly-stochastic matrix* is a matrix $\mathbf{Y} \in [0, 1]^{m \times m}$ for which the sum of every row, and the sum of every column, equals 1. That is $\sum_{j=1}^{m} y_{i,j} = 1$ for each i and $\sum_{i=1}^{m} y_{i,j} = 1$ for each j.

Permutation Matrices. Given some permutation $\pi\colon [m] \to [m]$, the *permutation matrix* $\mathbf{P}^\pi \in \{0,1\}^{m\times m}$ is the matrix where for each $(i,j) \in [m] \times [m]$, $P^\pi_{i,j}$ equals 1 if $i = \pi(j)$, and 0 otherwise.

Trivially, every permutation matrix is also a doubly-stochastic matrix. However, the BvN theorem sheds light on the other direction. It shows that every doubly-stochastic matrix can be obtained by a convex combination (a weighted sum with coefficients in $[0,1]$ which sum to 1) of all permutation matrices. Moreover, some of the constructive proofs of the theorem show how to find such a convex combination in which the number of nonzero coefficients is at most m^2.

Theorem 2 (BvN Theorem)**.** *Let $\mathbf{Y} \in [0,1]^{m\times m}$ be a doubly-stochastic matrix. We can decompose $\mathbf{Y}$ to a convex combination of at most m^2 permutation matrices, that is, we decompose $\mathbf{Y} = \lambda_1\mathbf{P}^{\tau_1} + \cdots + \lambda_q\mathbf{P}^{\tau_q}$ where each τ_t is a permutation with $\mathbf{P}^{\tau_t}$ being its corresponding permutation matrix, each $\lambda_t \in [0,1]$ and $\sum_{t=1}^{q}\lambda_t = 1$, and $q \leq m^2$. This decomposition can be found in polynomial time.*

See discussion and a classic proof to this form of the theorem in the appendix.

3. Algorithm for SR-WCM

In this section, we will devise approximation algorithms for SR-WCM by using a natural formulation of min-margin-SR-WCM as a linear program (LP). Let us first recall the problem; in min-margin-SR-WCM, we are given as input the scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ in the form of a vector $\boldsymbol{\alpha}$, a score profile σ, a designated preferred candidate $p \in \mathsf{C}$, and a dimension-k vector $\mathbf{w} = (w_\ell)_{\ell\in M}$ of positive integers representing the weights of a set M of k manipulators who will be added to the election. Specifically, the case where $\mathbf{w}$ is the all-ones vector defines the min-margin-SR-UCM problem. The goal is to minimize the maximum competitor score $\max_{c\in\mathsf{C}'} s(c)$, where $s(c')$ is c''s final score. Since the score margin is defined as $\max_{c\in\mathsf{C}'} s(c) - s(p)$, any additive approximation to the maximum competitor score is a same-factor additive approximation to the score margin. For the specific case of SR-UCM under a non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, we will later see how to convert this algorithm to a min-manipulator approximation algorithm, where our goal is to approximate k^{OPT}, the minimum number of manipulators required in order to enable p to win.

3.1 Linear Program for Min-Margin-SR-WCM

We can formulate the min-margin-SR-WCM as an integer program (IP). As solving IPs is NP-hard, we will relax it to the corresponding LP. As opposed to the LP for SR-UCM found in Keller et al.'s (2018) work, when switching to SR-WCM, because manipulators have different weights, the LP needs to be adapted to maintain the identity of the manipulators. Therefore, we define the variables $x_{i,j,\ell}$ for $(i,j,\ell) \in [m] \times [m] \times M$, with the intent that $x_{i,j,\ell}$ will equal 1 if competitor c_i receives the score α_j from manipulator ℓ, and 0 otherwise. We also define the variable T, with the intent that T will equal the maximum competitor score. The relaxed LP is defined as follows.

$$\min_{\mathbf{x},T} T$$

subject to:

$$\sum_{i=1}^{m} x_{i,j,\ell} = 1 \qquad \forall j \in [m], \ell \in M \ , \tag{1}$$

$$\sum_{j=1}^{m} x_{i,j,\ell} = 1 \qquad \forall i \in [m], \ell \in M \ , \tag{2}$$

$$\sigma(c_i) + \sum_{j \in [m], \ell \in M} w_\ell \alpha_j x_{i,j,\ell} \leq T \qquad \forall i \in [m] \ , \tag{3}$$

$$x_{i,j,\ell} \in [0,1] \qquad \forall i \in [m], j \in [m], \ell \in M \ , \tag{4}$$

where eq. (1) guarantees that every score type was awarded exactly once by each manipulator, eq. (2) guarantees that every competitor was awarded exactly one score by each manipulator, and eq. (3) guarantees that T upper-bounds the score of each competitor. When eq. (3) is combined with the minimization of T in the objective function, T should equal the maximum competitor score. The constraint $x_{i,j,\ell} \in [0,1]$ is a relaxation of the constraint $x_{i,j,\ell} \in \{0,1\}$, as the latter would have caused the LP to become an IP. We denote the relaxed LP as $\mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}, \sigma, \mathbf{w})$.

It should be noted that when treating the problem as a min-max problem, we need to take T as a variable that we wish to minimize (this is done by the objective function). However, if we consider the original definition in which our aim is to make the preferred candidate p win, T can be set to $\sigma(p) + k\alpha_0$ (the final score of p), and the LP will not have an objective function.

3.2 Rounding the Linear Program Solution

Assume that we solve the above LP, and let $(\mathbf{x}^\star, T^\star)$ be the resulting solution where $T^\star$ is the optimal objective value. As an optimum of the LP, $\mathbf{x}^\star$ denotes some allocation of scores such that the maximum competitor score is minimized. However, as this allocation is fractional, it does not translate into a valid strategy. Many algorithms work around such scenarios by using some form of *randomized rounding* over the fractional variables: roughly speaking, they try to round the values in $\mathbf{x}^\star$ randomly such that w.h.p. a good enough integral solution will be found. In our case, this seems quite problematic as the variables are highly interdependent, where their dependency is described by the LP constraints (1) and (2). These constraints should still hold even after the rounding.

To work around this, notice that each possible manipulator preference order—excluding p who will always be placed in the top position—corresponds to some permutation $\pi \colon [m] \to [m]$ (namely, $c_{\pi(j)}$ is ranked in the j-th position—or equivalently—c_i receives a score of $\alpha_{\pi^{-1}(i)}$). To maintain this property, randomized rounding should be done on the ballot level. However, to do so we have to define some distribution over the ballots (permutations), and there are $m!$ of them. This is where the BvN decomposition comes to the rescue.

Recall that M is the manipulator set. For some manipulator $\ell \in M$, observe the matrix $\mathbf{Y}^{(\ell)} = [y^{(\ell)}_{i,j}]_{(i,j) \in [m] \times [m]}$ where $y^{(\ell)}_{i,j} = x^\star_{i,j,\ell}$ and notice that it is *doubly-stochastic*, that

Algorithm 1: Min-margin-SR-WCM approximation algorithm

Solve $\mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}, \sigma, \mathbf{w})$, and let $(\mathbf{x}^\star, T^\star)$ be the resulting solution.
foreach *manipulator* $\ell \in M$ **do**
  Define $\mathbf{Y}^{(\ell)} = [y_{i,j}^{(\ell)}]_{i,j}$ where $y_{i,j}^{(\ell)} = x_{i,j,\ell}^\star$.
  Apply the BvN decomposition to $\mathbf{Y}^{(\ell)}$, and let $\Pi_\ell = \{\tau_1, \ldots, \tau_q\}$ be the resulting permutations with respective coefficients $\lambda_1, \ldots, \lambda_q$.
  Define a distribution $\hat{p}_\ell \colon \Pi_\ell \to [0,1]$ over Π_ℓ such that $\hat{p}_\ell(\tau_t) = \lambda_t$.
  Draw a random permutation $\pi_\ell \sim \hat{p}_\ell$.
  Define v_ℓ to be the preference order $p \succ c_{\pi_\ell(1)} \succ c_{\pi_\ell(2)} \succ \cdots \succ c_{\pi_\ell(m)}$, and assign it to ℓ as his ballot. `/* such that` ℓ `will award` $\alpha_{\pi_\ell^{-1}(i)}$ `to` c_i. `*/`
return *the resulting manipulator preference profile* $(v_\ell)_{\ell \in M}$.

is $\sum_i y_{i,j}^{(\ell)} = \sum_j y_{i,j}^{(\ell)} = 1$. Roughly speaking, the Birkhoff-von Neumann theorem states that each doubly-stochastic matrix is a point in the Birkhoff polytope, whose vertices are all the $m!$ permutation matrices. In other words, every doubly-stochastic matrix can be obtained by a convex combination of all permutation matrices. As mentioned, a convex combination in which the number of nonzero coefficients is at most m^2 can be efficiently found. A suitable variant of the Birkhoff-von Neumann theorem was given as Theorem 2. The remarkable thing about this form of the BvN decomposition, is that it implies that when choosing ballots for each of the manipulators, we only have to consider at most m^2 ballots—a polynomial number—out of the $m!$ possible ballots (that is, permutations of order m).

We proceed as follows. For each manipulator $\ell \in M$, consider $\mathbf{Y}^{(\ell)}$ and apply the BvN decomposition to it. Let $\Pi_\ell = \{\tau_1, \ldots, \tau_q\}$ (resp. $\lambda_1, \ldots, \lambda_q$) be the set of permutations (resp. coefficients) used in the above decomposition, and let $\hat{p}_\ell \colon \Pi_\ell \to [0,1]$ be a distribution over Π_ℓ such that $\hat{p}_\ell(\tau_t) = \lambda_t$. We draw a random permutation $\pi_\ell \sim \hat{p}_\ell$, and assign the preference order $p \succ c_{\pi_\ell(1)} \succ c_{\pi_\ell(2)} \succ \cdots \succ c_{\pi_\ell(m)}$ to ℓ as his ballot. Notice that as π_ℓ maps a 1-based rank j to some candidate index i, we can also say that c_i will be awarded a score of $\alpha_{\pi_\ell^{-1}(i)}$ by ℓ.

The above algorithm is summarized as Algorithm 1. To analyze it, we define $Q_i^\star = \sum_{j \in [m], \ell \in M} w_\ell \alpha_j x_{i,j,\ell}^\star$ to be the overall number of points awarded to c_i by the manipulators according to the fractional solution of the LP—as seen in Constraint (3). We also define $Q_{\max}^\star = \max_{i \in [m]} Q_i^\star$. In contrast, we let $\tilde{Q}_i = \sum_{\ell \in M} w_\ell \alpha_{\pi_\ell^{-1}(i)}$ denote the number of points awarded to c_i by the manipulators—following the BvN-based rounding. Recall that $R_1(\lambda, u, \mathbb{E}[X]) = 6\lambda \max\{\sqrt{u\mathbb{E}[X]}, u\} \ln \mathcal{N}$, where $\mathcal{N} = n|\mathsf{C}| = n(m+1)$. Let W_M be the sum of manipulator weights, and $w_{\max}^M$ be the maximum manipulator weight. In the next lemma we analyze $\tilde{Q}_i$ for each competitor c_i, specifically addressing the effect of the randomized rounding described by the algorithm.

Lemma 3. *With a failure probability of at most* $\mathcal{N}^{-\lambda+1}$*, at most* $R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star)$ *points will be added by Algorithm 1 to the score that each one of the competitors received according to the LP.*

Proof. According to the BvN decomposition, it holds that $\mathbf{Y}^{(\ell)} = \sum_{\pi \in \Pi_\ell} \hat{p}_\ell(\pi) \mathbf{P}^\pi$, where $\mathbf{P}^\pi$ is the permutation matrix corresponding to π. Therefore for each i, j:

$$y_{i,j}^{(\ell)} = \sum_{\pi \in \Pi_\ell} \hat{p}_\ell(\pi) \cdot P_{i,j}^\pi = \sum_{\substack{\pi \in \Pi_\ell \\ \pi(j)=i}} \hat{p}_\ell(\pi) \ .$$

Then:

$$\begin{aligned}
\mathbb{E}[\tilde{Q}_i] &= \sum_{\ell \in M} w_\ell \mathbb{E}_{\pi_\ell \sim \hat{p}_\ell}[\alpha_{\pi_\ell^{-1}(i)}] \\
&= \sum_{\ell \in M} w_\ell \sum_{\pi \in \Pi_\ell} \hat{p}_\ell(\pi) \alpha_{\pi^{-1}(i)} \\
&= \sum_{\ell \in M} w_\ell \sum_{j=1}^{m} \alpha_j \sum_{\substack{\pi \in \Pi_\ell \\ \pi(j)=i}} \hat{p}_\ell(\pi) \\
&= \sum_{\ell \in M} w_\ell \sum_{j=1}^{m} \alpha_j y_{i,j}^{(\ell)} \\
&= \sum_{j \in [m], \ell \in M} w_\ell \alpha_j x_{i,j,\ell}^\star \\
&\leq Q_{\max}^\star \ .
\end{aligned}$$

Applying Lemma 1 to $\tilde{Q}_i = \sum_{\ell \in M} w_\ell \alpha_{\pi_\ell^{-1}(i)}$ and recalling that $w_\ell \alpha_{\pi_\ell^{-1}(i)} \in [0, \alpha_1 w_{\max}^M]$, we get that with a failure probability of at most $\mathcal{N}^{-\lambda}$:

$$\begin{aligned}
\left| \tilde{Q}_i - \mathbb{E}[\tilde{Q}_i] \right| &\leq R_1(\lambda, \alpha_1 w_{\max}^M, \mathbb{E}[\tilde{Q}_i]) \\
&\leq R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star) \ .
\end{aligned}$$

In words, when we created valid ballots for each of the voters, the score of each competitor increased by at most $R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star)$ with a failure probability of at most $\mathcal{N}^{-\lambda}$. By applying the union-bound over all $m \leq \mathcal{N}$ competitors, this property can be made to hold for all competitors simultaneously with failure probability of at most $\mathcal{N}^{-\lambda+1}$. □

Corollary 4. *With a failure probability of at most $\mathcal{N}^{-\lambda+1}$, Algorithm 1 adds at most $R_1(\lambda, \alpha_1 w_{\max}^M, \alpha_1 W_M)$ points to the score each competitor received according to the LP.*

Proof. By combining Lemma 3 and the fact that $Q_{\max}^\star \leq \alpha_1 W_M$ since any (possibly fractional) allocation of points cannot exceed the case where all manipulators give the top score α_1 (we exclude α_0 since it will be awarded to p) to the same competitor. □

Recall that $T^\star$ is the (fractional) optimal objective value of the LP. In contrast, let T^{OPT} be the problem's (integral) optimum, i.e., the minimum highest competitor score, and let $\tilde{T} = \max_{i \in [m]} \sigma(c_i) + \tilde{Q}_i$ be the objective value obtained by our algorithm (following the randomized rounding). We are now ready to prove the main result of this section:

Theorem 5. *For SR-WCM under any score vector $\boldsymbol{\alpha}$, there exists a polynomial-time randomized algorithm yielding an*

$$\widetilde{O}(\min\{\alpha_1\sqrt{w_{\max}^M \cdot W_M}, \sqrt{\alpha_1 w_{\max}^M T^{\mathsf{OPT}}}\})$$

-additive approximation to the minimum highest competitor score (and thus to the margin minimization objective), with an exponentially-small failure probability.

Proof. For the approximation factor, Algorithm 1 adds at most $R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star)$ points to each candidate (by Lemma 3), and thus $\tilde{T} \leq T^\star + R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star) \leq T^{\mathsf{OPT}} + R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star)$. The last inequality holds since the LP is a relaxation of the original IP. From here, a rather straightforward analysis shows that $R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star) = \widetilde{O}(\sqrt{\alpha_1 w_{\max}^M T^{\mathsf{OPT}}})$ (since both $Q_{\max}^\star \leq T^\star \leq T^{\mathsf{OPT}}$ and $\alpha_1 w_{\max}^M \leq T^{\mathsf{OPT}}$). On the other hand, by employing Corollary 4, it holds that $R_1(\lambda, \alpha_1 w_{\max}^M, Q_{\max}^\star) = \widetilde{O}(\alpha_1\sqrt{w_{\max}^M \cdot W_M})$.

The overall running time is polynomial, as it is comprised of solving an LP (Karmarkar, 1984), followed by the polynomial-time BvN decomposition. The above algorithm has a polynomially-small failure probability $\mathcal{N}^{-\lambda+1}$. By choosing e.g. $\lambda = 2$, running it a linear number of times, and choosing the run yielding a minimal $\tilde{T}$, the failure probability becomes exponentially-small, while the runtime stays polynomial. □

3.3 Minimizing the Number of Manipulators

In this section, we shall provide results for one particularly interesting specific case, namely, that of min-manipulator-SR-UCM under non-concentrated scoring rules. Let us ignore the min-manipulator objective for a moment; Assume we are given *exactly* k manipulators, and let us discuss the score margin directly (as opposed to the maximum competitor score) this time. Let $g^{\mathsf{OPT}}(k)$ be the optimal score margin with exactly k manipulators, and likewise, let $g(k)$ be the score margin provided by our algorithm. Since an additive approximation to the maximum competitor score is also an additive approximation to the score margin, then for the specific case of SR-UCM, it holds that $g(k) \leq g^{\mathsf{OPT}}(k) + R_1(\lambda, \alpha_1, \alpha_1 k)$ by Corollary 4.

However, what if at this point we are willing to add more manipulators in order to close the gap? How many more manipulators do we require?

Let us first start with a slightly humbler task: how many more manipulators—besides the original k—do we need to add in order to reach a score margin of at most $g^{\mathsf{OPT}}(k)$? That is, we are willing to use more than k manipulators in order to reach the score margin that was optimal for *exactly* k manipulators.

Lemma 6 will answer this question and show that besides the k original manipulators, adding extra $\Delta(k) = \sqrt{k}\ln^3\mathcal{N}$ manipulators is sufficient, where—as before—$\mathcal{N} = n(m+1)$. This will provide a valuable insight that we will use later. To show this, in Lemma 6 we analyze the following suggested procedure. We call our min-margin-SR-UCM algorithm with k manipulators. We then add $\Delta(k)$ new manipulators; let M' be the set of these new manipulators. Each manipulator in M' will give p the top score and will rank all others randomly. Let $\bar{g}(k, \Delta(k))$ be the score margin obtained by this procedure, i.e., with k manipulators whose strategies were determined by our min-margin-SR-UCM algorithm, and $\Delta(k)$ extra manipulators with the strategy detailed above. In the following, we shall

assume w.l.o.g. that for a non-concentrated scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, it holds that $\bar{\alpha} \leq (1-2\epsilon)\alpha_0$ for some constant $0 < \epsilon < 1/2$.

Lemma 6. *Let $\mathcal{R}_{\boldsymbol{\alpha}}$ be a scoring rule. With a failure probability of at most $2\mathcal{N}^{-\lambda+1}$, it holds that $\bar{g}(k, \Delta(k)) \leq g^{\mathsf{OPT}}(k)$.*

Proof. Since $g(k) \leq g^{\mathsf{OPT}}(k) + R_1(\lambda, \alpha_1, \alpha_1 k)$, it is enough to show that $\bar{g}(k, \Delta(k)) \leq g(k) - R_1(\lambda, \alpha_1, \alpha_1 k)$, or in words, that adding the $\Delta(k)$ extra manipulators reduces the score margin by at least $R_1(\lambda, \alpha_1, \alpha_1 k)$.

Recall that C' is the set of candidates excluding p and let $s'(c)$ be the score of a candidate c *before* the addition of the $\Delta(k)$ extra manipulators. Fix a candidate $c \in \mathsf{C}'$ and let $g_c = s'(c) - s'(p)$ be the margin between p and c just before adding the $\Delta(k)$ extra manipulators. Let $D = \sum_{\ell \in M'} (\alpha_0 - \alpha_{v_\ell(c)}) = \Delta(k)\alpha_0 - \sum_{\ell \in M'} \alpha_{v_\ell(c)}$, be the decrease in g_c as a result of the new manipulators in M', where $\alpha_{v_\ell(c)}$ is the (random) score ℓ awards c. Notice that D is the sum of independent random variables, that $\mathbb{E}[D] = \Delta(k)(\alpha_0 - \bar{\alpha}) \geq 2\Delta(k)\epsilon\alpha_0$ and that $\mathbb{E}[D] = \Delta(k)(\alpha_0 - \bar{\alpha}) < \Delta(k)\alpha_0$. Applying Lemma 1, with a failure probability of at most $\mathcal{N}^{-\lambda}$, we obtain:

$$\begin{aligned} D &\geq \mathbb{E}[D] - R_1(\lambda, \alpha_0, \mathbb{E}[D]) \\ &\geq 2\Delta(k)\epsilon\alpha_0 - R_1(\lambda, \alpha_0, \Delta(k)\alpha_0) \qquad (\text{since } 2\Delta(k)\epsilon\alpha_0 \leq \mathbb{E}[D] \leq \Delta(k)\alpha_0) \end{aligned}$$

Observe $\Delta(k)$ and notice that it is defined so that $R_1(\lambda, \alpha_1, \alpha_1 k) = O(\alpha_0 \Delta(k)/\log^2 \mathcal{N})$ and also $R_1(\lambda, \alpha_0, \alpha_0 \Delta(k)) = O(\alpha_0 \Delta(k)/\log^{1/2} \mathcal{N})$. Therefore $R_1(\lambda, \alpha_1, \alpha_1 k) = o(\alpha_0 \Delta(k))$ and also $R_1(\lambda, \alpha_0, \alpha_0 \Delta(k)) = o(\alpha_0 \Delta(k))$. We can assume w.l.o.g. that our $\mathcal{N}$ is large enough such that both $R_1(\lambda, \alpha_1, \alpha_1 k) < \epsilon\alpha_0 \Delta(k)$ and $R_1(\lambda, \alpha_0, \alpha_0 \Delta(k)) < \epsilon\alpha_0 \Delta(k)$, as otherwise $\mathcal{N}$ is constant and the entire problem can be easily solved in constant time. Therefore $D \geq 2\Delta(k)\epsilon\alpha_0 - R_1(\lambda, \alpha_0, \Delta(k)\alpha_0) \geq \Delta(k)\epsilon\alpha_0 \geq R_1(\lambda, \alpha_1, \alpha_1 k)$. In words, the margin between any competitor c and p had decreased by at least $R_1(\lambda, \alpha_1, \alpha_1 k)$. In particular, the score margin had decreased by at least $R_1(\lambda, \alpha_1, \alpha_1 k)$

The failure probability is $2\mathcal{N}^{-\lambda+1}$ by a union-bound over one use of Lemma 1 for each of the competitors, in addition to the failure probability of Corollary 4. □

We will now assume that we get an instance of min-manipulator-SR-UCM under a non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, and that the optimal number of manipulators needed for p to win is k^{OPT}. Assume for a moment that we have some oracle telling us what is the value of k^{OPT}, but not providing us with the optimal strategy. We can use the procedure defined above and Lemma 6 to close the score margin and let p win. This is further elaborated by the following theorem.

Theorem 7. *Let $\mathcal{R}_{\boldsymbol{\alpha}}$ be a scoring rule. For SR-UCM under $\mathcal{R}_{\boldsymbol{\alpha}}$, let k^{OPT} be the minimum number of manipulators required to make p win. Then there exists a polynomial-time randomized algorithm finding a manipulator strategy making p win using at most $k^{\mathsf{OPT}} + \widetilde{O}(\sqrt{k^{\mathsf{OPT}}})$ manipulators, with an exponentially-small failure probability.*

Proof. Following the above discussion, we can try every $k = 1, 2, \ldots,$ as our initial guess of k^{OPT}, continue with the rest of the steps described below, and finally stick with the minimum overall number of manipulators $\tilde{k} = k + \Delta(k)$ for which p wins. A concrete bound on the number of iterations will be detailed below.

Consider the specific iteration where $k = k^{\mathsf{OPT}}$; p can hypothetically win (given a good enough strategy) and therefore $g^{\mathsf{OPT}}(k) \leq 0$. According to Lemma 6, $\bar{g}(k, \Delta(k)) \leq g^{\mathsf{OPT}}(k) \leq 0$. Therefore, by using $\tilde{k} = k + \Delta(k) = k^{\mathsf{OPT}} + \Delta(k^{\mathsf{OPT}}) = k^{\mathsf{OPT}} + \widetilde{O}(\sqrt{k^{\mathsf{OPT}}})$ manipulators as described above, we make p win.

For a single guess of k, the failure probability is $2\mathcal{N}^{-\lambda+1}$ according to Lemma 6. Setting $\lambda = 2$ is enough to make the failure probability $1/\Omega(\mathcal{N})$. By repeating this a linear number of times (each time checking if p wins), the failure probability becomes exponentially-small, while the runtime stays polynomial.

Now consider the outer loop over k; with an exponentially-small failure probability it will find a solution when $k = k^{\mathsf{OPT}}$ (and possibly before; it is possible that when $k < k^{\mathsf{OPT}}$, a solution will still be found because of the $\Delta(k)$ extra manipulators) and therefore at most k^{OPT} iterations are sufficient. Under this k^{OPT}-multiplicative factor, the failure probability remains exponentially-small, while the runtime remains polynomial. □

4. Algorithm for SR-Weighted-$Bribery

In this section, we will present approximation algorithms for Bribery and its weighted and variable-price generalizations. We will focus on two broad families of scoring rules: constant scoring rules, and non-concentrated scoring rules.

Recall that an instance of Weighted-$Bribery is comprised of a weighted election E (represented by the candidate set C, the preference profile V, and the weight vector $\mathbf{w}$), a scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ (represented by the score vector $\boldsymbol{\alpha}$), a price ψ_ℓ for each voter ℓ, and a preferred candidate p. As mentioned before, a scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$ is called *constant* if all the values in $\boldsymbol{\alpha}$ are integers, and $\alpha_0 = O(1)$. A scoring rule is called *non-concentrated* if $\bar{\alpha} \leq (1-\epsilon)\alpha_0$, for some constant $\epsilon > 0$, where $\bar{\alpha} = 1/m \cdot \sum_{j=1}^{m} \alpha_j$ is the average of the values in $\boldsymbol{\alpha}$ excluding α_0.

As part of our (approximate) solutions to these problems, we will rely on our solution for min-margin-SR-WCM from Section 3 (whose approximation guarantees will better suit our needs than the one by Keller et al., 2019). Similar to our SR-WCM algorithm, the algorithm for SR-Bribery will also rely on an LP. However, this LP will be slightly more involved.

4.1 LP for SR-Weighted-$Bribery

Let y_ℓ be an indicator variable for each voter ℓ indicating whether he should be bribed, and let W be a variable denoting the overall weight of the bribed voters. We also need to describe how to allocate the points of the manipulator ballots. Therefore—similarly to what we did for SR-WCM—we define the variables $x_{i,j,\ell}$ for $(i,j,\ell) \in [m] \times [m] \times N$, with the intent that $x_{i,j,\ell}$ will equal 1 if candidate c_i receives a score α_j from voter ℓ after he is bribed, and 0 otherwise. (Notice that we have not defined the variables $x_{i,j,\ell}$ for $j = 0$, since when we bribe a voter, he will always award α_0 to p.) In contrast to the SR-WCM LP, the main issue is to guarantee that for a voter ℓ, the $x_{i,j,\ell}$ variables have no meaning unless ℓ is one of the bribed voters. Our formulation of the LP for SR-Weighted-$Bribery will show how to solve this. We first define an IP, and relax it later.

$$\min_{\mathbf{x},W,\mathbf{y},T,\Psi} \Psi$$

subject to:

$$\sum_{\ell \in N} \psi_\ell y_\ell = \Psi \tag{5}$$

$$\sum_{\ell \in N} y_\ell w_\ell = W \tag{6}$$

$$\sigma(c_i) - \sum_{\ell \in N} y_\ell w_\ell \alpha_{v_\ell(c_i)} + \sum_{\ell \in N, j \in [m]} w_\ell \alpha_j x_{i,j,\ell} \leq T \qquad \forall i \in [m] \ , \tag{7}$$

$$\sigma(p) - \sum_{\ell \in N} y_\ell w_\ell \alpha_{v_\ell(p)} + \alpha_0 W \geq T \tag{8}$$

$$\sum_{i=1}^{m} x_{i,j,\ell} = y_\ell \qquad \forall j \in [m], \ell \in N \ , \tag{9}$$

$$\sum_{j=1}^{m} x_{i,j,\ell} = y_\ell \qquad \forall i \in [m], \ell \in N \ , \tag{10}$$

$$y_\ell \in \{0,1\} \qquad \forall \ell \in N \ , \tag{11}$$

$$x_{i,j,\ell} \in \{0,1\} \qquad \forall i \in [m], j \in [m], \ell \in N \ . \tag{12}$$

For each voter ℓ, Constraints (9) and (10) make sure that this is a valid allocation if ℓ is bribed (and therefore $y_\ell = 1$), and otherwise make sure that $x_{i,j,\ell} = 0$ for all $i \in [m], j \in [m]$. Constraint (5) ensures that the overall price of the bribed voters will be Ψ, the variable we wish to minimize, while Constraint (6) defines their aggregate weight. Recall that by $v_\ell(c)$ for some voter ℓ and some candidate c, we denote the 0-based position of c in ℓ's preference order. Thus, $\alpha_{v_\ell(c)}$ is the score currently given by a voter ℓ to a candidate c. Constraints (7) and (8) together make sure that p has a final score greater than or equal to any other candidate. Notice that the use of $\alpha_0 W$ in eq. (8) stems from the fact that the score awarded to p by the manipulators is known; each will give her the maximum score available, α_0. The two last constraints ensure that the y_ℓ and $x_{i,j,\ell}$ variables are indeed indicator variables.

As usual, as solving an IP is NP-hard, we relax it into an LP by replacing the integrality constraints $y_\ell \in \{0,1\}$ and $x_{i,j,\ell} \in \{0,1\}$ (for all (i,j,ℓ)) with the corresponding $y_\ell \in [0,1]$ and $x_{i,j,\ell} \in [0,1]$. Let the above LP be denoted by LP_{B}.

4.2 Algorithm Outline

Our algorithm is described as Algorithm 2, and is comprised of four stages. In the first stage, we shall solve LP_{B}. In the second, we will choose an initial set of voters to bribe using a simple form of randomized rounding. This choice of voters will reduce the problem to an instance of the min-margin-SR-WCM problem, in which the manipulators—and their weights—are known (as we have already determined them). This is the point—the third stage—where we shall use our min-margin-SR-WCM algorithm to determine their strategy. Our main claim here is that by bribing these voters without spending too much

Algorithm 2: SR-Weighted-$Bribery approximation algorithm

Solve LP_{B} as described, and let $(\mathbf{x}^\star, \mathbf{y}^\star, W^\star, T^\star, \Psi^\star)$ be the resulting solution.
foreach $\ell \in N$ **do**
  $\tilde{y}_\ell \leftarrow \begin{cases} 1 & \text{with probability } y^\star_\ell; \\ 0 & \text{otherwise.} \end{cases}$
Let $M' = \{\, \ell \mid \tilde{y}_\ell = 1 \,\}$ and $\tilde{\mathbf{w}} = (w_\ell)_{\ell \in M'}$.
Define $\hat{\sigma}(c_i) = \sigma(c_i) - \sum_{\ell \in N} \alpha_{v_\ell(c_i)} \tilde{y}_\ell$ for every $i \in [m]$.
Apply our min-margin-SR-WCM algorithm on the input $((\alpha_1, \ldots, \alpha_m), \hat{\sigma}, \tilde{\mathbf{w}})$, and update M''s preference orders accordingly.
if *p is still losing to the highest-scoring competitor* **then**
  Bribe additional voters according to either Algorithm 3 (constant scoring rules), or Algorithm 4 (non-concentrated scoring rules). Let M'' be the set of these additional bribed voters.
Let $M = M' \cup M''$ be the set of all voters we have bribed.
return *the identity of the bribed voters M and their new preference orders.*

in the process, and assigning them ballots, we have reduced this SR-Weighted-$Bribery instance to another SR-Weighted-$Bribery instance, in which the margin is relatively small. Then, at the fourth stage, we will show that this margin is relatively easy to close, by bribing more voters, spending a relatively small additional price in the process.

4.3 Stage 1: Solving the SR-Weighted-$Bribery LP

We solve LP_{B} using a polynomial-time solver (Karmarkar, 1984) and obtain the solution $(\mathbf{x}^\star, \mathbf{y}^\star, W^\star, T^\star, \Psi^\star)$ where $\Psi^\star$ is the optimal objective value. While $(\mathbf{x}^\star, \mathbf{y}^\star, W^\star, T^\star, \Psi^\star)$ is a solution to a fractional version of SR-Weighted-$Bribery (due to our use of a relaxed LP), it will enable us to obtain an *integral* solution to SR-Weighted-$Bribery without too much compromise on the increase in the overall price paid, or *cost*.

4.4 Stage 2: Rounding $\mathbf{y}^\star$

Observe the vector $\mathbf{y}^\star$ which is part of the solution of LP_{B}. As its values are fractional, $y^\star_\ell$ does not describe whether ℓ should be bribed or not. To address this, we shall round $\mathbf{y}^\star$ without touching $\mathbf{x}^\star$ for now. This is done by defining a new vector $\tilde{\mathbf{y}} = (\tilde{y}_\ell)_{\ell \in N}$, such that:

$$\tilde{y}_\ell = \begin{cases} 1 & \text{with probability } y^\star_\ell; \\ 0 & \text{otherwise.} \end{cases}$$

By rounding $\mathbf{y}^\star$ into $\tilde{\mathbf{y}}$, we have determined a set of bribed voters $M' = \{\, \ell \mid \tilde{y}_\ell = 1 \,\}$. Now let $\Psi_{M'} = \sum_{\ell \in N} \psi_\ell \tilde{y}_\ell = \sum_{\ell \in M'} \psi_\ell$ and $W_{M'} = \sum_{\ell \in N} w_\ell \tilde{y}_\ell = \sum_{\ell \in M'} w_\ell$, and notice that $\Psi_{M'}$ and $W_{M'}$ are the counterparts of $\Psi^\star$ and $W^\star$, respectively, when substituting $\tilde{\mathbf{y}}$ for $\mathbf{y}^\star$ in their respective formulas in LP_{B}. Recall that $R_1(\lambda, u, \mathbb{E}[X]) = 6\lambda \max\{\sqrt{u\mathbb{E}[X]}, u\} \ln \mathcal{N}$, where $\mathcal{N} = n(m+1)$; that the value $T^\star$—which is part of the solution of LP_{B}—bounds the score (according to LP_{B}) of each competitor from above on one hand, and on the other hand,

bounds the score (again, according to LP_{B}) of p from below; that $w_{\max}$ is the maximum voter weight, and $\psi_{\max}$ is the maximum voter price.

Lemma 8. *The following statements are true:*

1. *With a probability of at least* $1-\mathcal{N}^{-\lambda}$,

$$\Psi_{M'} \in [\Psi^\star \pm R_1(\lambda, \psi_{\max}, \Psi^\star)] \ ;$$

2. *With a probability of at least* $1-\mathcal{N}^{-\lambda}$,

$$W_{M'} \in [W^\star \pm R_1(\lambda, w_{\max}, W^\star)] \ ;$$

3. *For every* $c \in \mathsf{C}'$, *with a probability of at least* $1-\mathcal{N}^{-\lambda}$,

$$\sigma(c_i) - \sum_{\ell \in N} \tilde{y}_\ell w_\ell \alpha_{v_\ell(c_i)} + \sum_{\ell \in N, j \in [m]} w_\ell \alpha_j x^\star_{i,j,\ell} \leq T^\star + \alpha_0 R_1(\lambda, w_{\max}, W^\star) \ ;$$

4. *With a probability of at least* $1-\mathcal{N}^{-\lambda}$,

$$\sigma(p) - \sum_{\ell \in N} \tilde{y}_\ell w_\ell \alpha_{v_\ell(p)} + \alpha_0 W^\star \geq T^\star - \alpha_0 R_1(\lambda, w_{\max}, W^\star) \ .$$

Proof. Notice the following observations:

- $\Psi_{M'} = \sum_{\ell \in N} \psi_\ell \tilde{y}_\ell$ is a sum of independent random variables with an expected value $\Psi^\star$, where each summand is in $[0, \psi_{\max}]$.
- $W_{M'} = \sum_{\ell \in N} w_\ell \tilde{y}_\ell$ is a sum of independent random variables with an expected value $W^\star$, where each summand is in $[0, w_{\max}]$.
- $\sum_{\ell \in N} \tilde{y}_\ell w_\ell \alpha_{v_\ell(c)}$ is a sum of independent random variables with an expected value $\sum_{\ell \in N} y^\star_\ell w_\ell \alpha_{v_\ell(c)} \leq \alpha_0 W^\star$, where each summand is in $[0, \alpha_0 w_{\max}]$.

Applying Lemma 1 to the first two observations yields the first two statements. Before handling the other two statements, notice that $R_1(\lambda, \alpha_0 w_{\max}, \alpha_0 W^\star) \leq \alpha_0 R_1(\lambda, w_{\max}, W^\star)$. Using this when applying Lemma 1 to the latter two observations, and bringing into consideration eq. (7) and eq. (8) respectively, yields the last two statements. □

4.5 Stage 3: Running the SR-WCM Algorithm

Recall that in the Weighted-$Bribery LP which we denoted as LP_{B}, the y_ℓ variables describe the identity of the bribed voters, the $x_{i,j,\ell}$ variables describe their new ballots, W is the bribed voters' overall weight, Ψ is the bribed voters' overall price, and finally T upper-bounds each competitor score and lower-bounds p's score. We have solved LP_{B}, obtaining the values $(\mathbf{x}^\star, \mathbf{y}^\star, W^\star, T^\star, \Psi^\star)$ for these variables; however, these values are fractional and thus do not describe a concrete decision of the identity of bribed voters and their new ballots. In the previous section we have started addressing that; we rounded the values in

$\mathbf{y}^\star$—obtaining $\tilde{\mathbf{y}}$ in the process—and accordingly adjusted $\Psi^\star$ to become $\Psi_{M'}$ and $W^\star$ to become $W_{M'}$. In this section, we will find a way to also round $\mathbf{x}^\star$.

Recall that by $v_\ell(c)$ for some voter ℓ and some candidate c, we denote the 0-based position of c in ℓ's original preference order. Let $\boldsymbol{\alpha}' = (\alpha_1, \ldots, \alpha_m)$, and let $\hat{\sigma}(c_i) = \sigma(c_i) - \sum_{\ell \in N} \tilde{y}_\ell w_\ell \alpha_{v_\ell(c_i)}$ for $i = 1, \ldots, m$. In words, $\hat{\sigma}(c_i)$ is $\sigma(c_i)$ when it is adjusted for the loss of the voters who were 'deleted' as described by the vector $\tilde{\mathbf{y}}$ (we use the term "deleted" as bribing a voter can be seen as deleting him and then adding a manipulator having the same weight). Notice that $\hat{\sigma}$ is defined only for the competitors. We have now reduced the problem to the min-margin-SR-WCM problem: $\hat{\sigma}$ is the new score profile, $M' = \{\, \ell \mid \tilde{y}_\ell = 1 \,\}$ are the manipulators (one for each of the deleted voters), $\tilde{\mathbf{w}} = (w_\ell)_{\ell \in M'}$ is their respective weight-vector (of dimension $|M'|$, as opposed to $\mathbf{w}$, which was of dimension $|N| = n$), $W_{M'} = \sum_{\ell \in M'} w_\ell$ and $\boldsymbol{\alpha}'$ is $\boldsymbol{\alpha}$ without the score α_0, which is always awarded to p by the manipulators, and thus is irrelevant to the input.

We apply our min-margin-SR-WCM algorithm on $(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$. Here we do not rely on its approximation guarantee as provided by Theorem 5, but on the stronger Corollary 4, showing that the resulting $R_1(\lambda, \alpha_1 w_{\max}^{M'}, \alpha_1 W_{M'})$-factor is an additive term not just w.r.t. the min-margin-SR-WCM optimum, but also w.r.t. the fractional solution of $\mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$.

Recall that in both our LPs, for SR-WCM ($\mathsf{LP}_{\mathrm{CM}}$) and for Weighted-$Bribery (LP_{B}) we have defined a variable T which upper-bounds the maximum competitor score. In this spirit, let $\mathcal{L}$ be a shorthand to $\mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$, and let $T^\star_{\mathcal{L}}$ be the optimal objective value of $\mathcal{L}$. Also let $\tilde{T}$ be the maximum candidate score as a result of our min-margin-SR-WCM algorithm on the input $(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$. We are interested in comparing the three different values for T that we have encountered hitherto:

- $T^\star$, i.e., the value of T when solving LP_{B}. Here, both y and x variables are fractional.
- $T^\star_{\mathcal{L}}$, i.e., the value of T when solving $\mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$ (which is done as part of our SR-WCM algorithm). Here, we have already decided who to bribe, but after solving $\mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$ the manipulator strategies are still fractional.
- $\tilde{T}$, maximum competitor score obtained by fully running our min-margin-SR-WCM algorithm on $(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$. After running this algorithm, we have an integral manipulation strategy for the voters in M'.

We wish to show that the overall difference between $T^\star$ and $\tilde{T}$ is relatively small, and we will do this by comparing them to $T^\star_{\mathcal{L}}$. This will be shown by Lemma 11, but first, in Lemma 9 and corollary 10 we will compare $T^\star_{\mathcal{L}}$ and $T^\star$.

Let $R_2 = \alpha_0 R_1(\lambda, w_{\max}, W^\star) + \alpha_1 R_1(\lambda, w_{\max}, W^\star)$. Then:

Lemma 9. *With a failure probability of at most $2\mathcal{N}^{-\lambda+1}$, there is a (not necessarily optimal) solution $\mathbf{x}' = [x'_{i,j,\ell}]_{(i,j,\ell) \in [m] \times [m] \times M'}$ to $\mathcal{L}$ with an objective value T' such that $T' \leq T^\star + R_2$.*

Proof. Define $x'_{i,j,\ell} = z_{i,j}$ where

$$z_{i,j} = \frac{\sum_{\ell' \in N} w_{\ell'} x^\star_{i,j,\ell'}}{W^\star}$$

for all i, j and $\ell \in M'$. Notice that a division by 0 in the equation cannot occur since w.l.o.g. $W^\star > 0$ (the opposite case implies that in LP_{B}, $y^\star_\ell = 0$ for all $\ell \in N$ and this implies that

p was winning in the first place). Notice that in $\mathsf{LP}_{\mathrm{CM}}$, eqs. (1) and (2) immediately hold w.r.t. $\mathbf{x}'$, as:

$$\sum_{i=1}^{m} x'_{i,j,\ell} = \frac{\sum_{i=1}^{m}\sum_{\ell'\in N} w_{\ell'} x^\star_{i,j,\ell'}}{W^\star} = \frac{\sum_{\ell'\in N} w_{\ell'}\sum_{i=1}^{m} x^\star_{i,j,\ell'}}{W^\star} = \frac{\sum_{\ell'\in N} w_{\ell'} y^\star_{\ell'}}{W^\star} = 1 \ , \qquad (13)$$

for all $j \in [m], \ell \in M'$ and

$$\sum_{j=1}^{m} x'_{i,j,\ell} = \frac{\sum_{j=1}^{m}\sum_{\ell'\in N} w_{\ell'} x^\star_{i,j,\ell'}}{W^\star} = \frac{\sum_{\ell'\in N} w_{\ell'}\sum_{j=1}^{m} x^\star_{i,j,\ell'}}{W^\star} = \frac{\sum_{\ell'\in N} w_{\ell'} y^\star_{\ell'}}{W^\star} = 1 \ , \qquad (14)$$

for all $i \in [m], \ell \in M'$. In both eqs. (13) and (14), the third equality follows from the satisfied LP_{B} constraints, eqs. (9) and (10), respectively. As a direct corollary of eq. (14), notice that

$$\sum_{j=1}^{m} z_{i,j} = \sum_{j=1}^{m} x'_{i,j,\ell} = 1 \ . \qquad (15)$$

For all i, with a failure probability to be determined, it holds that the number of points awarded to c_i by the voters in M', that is, $\sum_{j\in[m],\ell\in M'} w_\ell \alpha_j x'_{i,j,\ell}$, satisfies:

$$\begin{aligned}
&\sum_{j\in[m],\ell\in M'} w_\ell \alpha_j x'_{i,j,\ell} \\
&\quad= \sum_{j\in[m],\ell\in M'} w_\ell \alpha_j z_{i,j} && \text{(by definition of } \mathbf{x}') \\
&\quad\le \left(\sum_{\ell\in M'} w_\ell\right)\left(\sum_{j=1}^{m} \alpha_j z_{i,j}\right) \\
&\quad= W_{M'}\sum_{j=1}^{m} \alpha_j z_{i,j} \\
&\quad= W^\star \sum_{j=1}^{m} \alpha_j z_{i,j} + (W_{M'} - W^\star)\sum_{j=1}^{m} \alpha_j z_{i,j} \\
&\quad= W^\star \left(\sum_{j=1}^{m} \alpha_j \frac{\sum_{\ell'\in N} w_{\ell'} x^\star_{i,j,\ell'}}{W^\star}\right) + (W_{M'} - W^\star)\sum_{j=1}^{m} \alpha_j z_{i,j} \\
&\quad= \sum_{j\in[m],\ell'\in N} w_{\ell'} \alpha_j x^\star_{i,j,\ell'} + (W_{M'} - W^\star)\sum_{j=1}^{m} \alpha_j z_{i,j} \\
&\quad\le \sum_{j\in[m],\ell'\in N} w_{\ell'} \alpha_j x^\star_{i,j,\ell'} + R_1(\lambda, w_{\max}, W^\star)\sum_{j=1}^{m} \alpha_j z_{i,j} && \text{(by Lemma 8.2)} \\
&\quad\le \sum_{j\in[m],\ell'\in N} w_{\ell'} \alpha_j x^\star_{i,j,\ell'} + R_1(\lambda, w_{\max}, W^\star)\cdot \alpha_1 \sum_{j=1}^{m} z_{i,j} \\
&\quad\le \sum_{j\in[m],\ell'\in N} w_{\ell'} \alpha_j x^\star_{i,j,\ell'} + \alpha_1 R_1(\lambda, w_{\max}, W^\star) && \text{(by eq. (15))} \ .
\end{aligned}$$

To summarize:

$$\sum_{j\in[m],\ell\in M'} w_\ell \alpha_j x'_{i,j,\ell} \leq \sum_{j\in[m],\ell'\in N} w_{\ell'} \alpha_j x^\star_{i,j,\ell'} + \alpha_1 R_1(\lambda, w_{\max}, W^\star) \ . \tag{16}$$

Recall that the score profile $\hat{\sigma}$, where $\hat{\sigma}(c_i) = \sigma(c_i) - \sum_{\ell\in N} \tilde{y}_\ell w_\ell \alpha_{v_\ell(c_i)}$ for each $i = 1,\ldots,m$, was the score profile with which we called our min-margin-SR-WCM algorithm. Plugging eq. (16) into eq. (3), it then follows that:

$$\begin{aligned}
&\hat{\sigma}(c_i) + \sum_{j\in[m],\ell\in M'} w_\ell \alpha_j x'_{i,j,\ell} \\
&\quad = \sigma(c_i) - \sum_{\ell\in N} w_\ell \alpha_{v_\ell(c_i)} \tilde{y}_\ell + \sum_{j\in[m],\ell\in M'} w_\ell \alpha_j x'_{i,j,\ell} && \text{(by definition of } \hat{\sigma}) \\
&\quad \leq \sigma(c_i) - \sum_{\ell\in N} w_\ell \alpha_{v_\ell(c_i)} \tilde{y}_\ell + \sum_{j\in[m],\ell'\in N} w_{\ell'} \alpha_j x^\star_{i,j,\ell'} \\
&\qquad + \alpha_1 R_1(\lambda, w_{\max}, W^\star) && \text{(by eq. (16))} \\
&\quad \leq T^\star + \alpha_0 R_1(\lambda, w_{\max}, W^\star) \\
&\qquad + \alpha_1 R_1(\lambda, w_{\max}, W^\star) && \text{(by Lemma 8.3)} \\
&\quad = T^\star + R_2 \ ,
\end{aligned}$$

The above holds with a failure probability of $2\mathcal{N}^{-\lambda}$ stemming from the two uses of Lemma 8 (each contributing $\mathcal{N}^{-\lambda}$ to the failure probability). Since we would like the above to hold for all i simultaneously, the failure probability becomes at most $2\mathcal{N}^{-\lambda+1}$ by the union bound. Therefore, we have just shown that w.h.p. eq. (3) holds w.r.t. the objective value $T' = \max_{i\in[m]}(\hat{\sigma}(c_i) + \sum_{j\in[m],\ell\in M'} w_\ell \alpha_j x'_{i,j,\ell}) \leq T^\star + R_2$.

To recap, we have now defined a valid solution to $\mathcal{L}$ with an objective value of at most $T^\star + R_2$ with a high probability. □

Recall that $R_2 = \alpha_0 R_1(\lambda, w_{\max}, W^\star) + \alpha_1 R_1(\lambda, w_{\max}, W^\star)$. In the last lemma we have shown the existence of a (not necessarily optimal) solution to the resulting SR-WCM LP instance $\mathcal{L} = \mathsf{LP}_{\mathrm{CM}}(\boldsymbol{\alpha}', \hat{\sigma}, \tilde{\mathbf{w}})$ having an objective value T' such that $T' \leq T^\star + R_2$.

Corollary 10. *With a failure probability of at most $2\mathcal{N}^{-\lambda+1}$, it holds that $T^\star_{\mathcal{L}} \leq T^\star + R_2$.*

Proof. This is because $T^\star_{\mathcal{L}} \leq T'$, as the optimal solution cannot be worse than the solution we defined in the previous lemma. □

Recall that $\tilde{T}$ is the value of T after running our min-margin-SR-WCM algorithm, which supplied us with an integral manipulation strategy for the voters in M'. Define:

$$R_3 = R_2 + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1(W^\star + R_1(\lambda, w_{\max}, W^\star))) \ .$$

Lemma 11. *With a probability of at least $1 - 4\mathcal{N}^{-\lambda+1}$, it holds that $\tilde{T} \leq T^\star + R_3$.*

Proof. By combining Corollary 4 w.r.t. $\tilde{T}$ and $T^\star_{\mathcal{L}}$, and Corollary 10, we obtain that, with a failure probability of at most $4\mathcal{N}^{-\lambda+1}$:

$$\begin{aligned}
\tilde{T} &\leq T^\star_{\mathcal{L}} + R_1(\lambda, \alpha_1 w^{M'}_{\max}, \alpha_1 W_{M'}) && \text{(by Corollary 4)}\\
&\leq T^\star_{\mathcal{L}} + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 W_{M'})\\
&\leq T^\star_{\mathcal{L}} + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 (W^\star + R_1(\lambda, w_{\max}, W^\star))) && \text{(by Lemma 8.2)}\\
&\leq T^\star + R_2\\
&\qquad + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 (W^\star + R_1(\lambda, w_{\max}, W^\star))) && \text{(by Corollary 10)}\\
&= T^\star + R_3 \ ,
\end{aligned}$$

The aforementioned failure probability is obtained by a union-bound over the failure probability of each of the lemmas used. □

4.6 Stage 4: Bribing More Voters

Recall that by now we have the identity of the bribed voters encoded as the binary vector $\tilde{\mathbf{y}}$, and also represented as the set $M' = \{\, \ell \mid \tilde{y}_\ell = 1 \,\}$. We also have an integral strategy (i.e., ballots) for them obtained by running our min-margin-SR-WCM algorithm. Let $\tilde{\mathbf{x}} = [\tilde{x}_{i,j,\ell}]$ be the indicator variables describing the allocation of the scores to the candidates according to our min-margin-SR-WCM algorithm (so for every $\ell \in M'$, $\tilde{x}_{i,j,\ell}$ equals 1 if candidate c_i receives a score of type α_j from ℓ, and 0 otherwise). $\tilde{\mathbf{x}}$ is merely an alternative way to describe the new preference orders of the voters in M'. As mentioned, we denote the maximum competitor score induced by $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ as $\tilde{T}$. At this point, we supposedly have a valid, integral strategy: $\tilde{\mathbf{y}}$ describes who to bribe and $\tilde{\mathbf{x}}$ describes the *new* preference order of each of the bribed voters. Does the bribery-scheme and strategy as described by $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$ constitute a valid solution to SR-Weighted-$Bribery? The answer is unfortunately no. Let $s'(c)$ be the current score of a candidate c. While $\tilde{T}$ bounds the score $s'(c)$ of each competitor c, p's current score $s'(p) = \sigma(p) - \sum_{\ell \in N} w_\ell \alpha_{v_\ell(p)} \tilde{y}_\ell + \alpha_0 W_{M'}$ might be less than $\tilde{T}$. In other words, Constraint (8) of LP_{B} might not hold w.r.t. $\tilde{\mathbf{y}}$, $W_{M'}$, and $\tilde{T}$. However, in the following lemma we show that the margin needed for Constraint (8) to hold is not too large; let $R_4 = 2\alpha_0 R_1(\lambda, w_{\max}, W^\star) + R_3$.

Lemma 12. *With a probability of at least* $1 - 6\mathcal{N}^{-\lambda+1}$*, it holds that* $s'(p) \geq \tilde{T} - R_4$.

Proof. Recall that $T^\star$ is the bound on the maximum competitor score in the (fractional) solution of LP_{B}. Then assuming none of the previous lemmas fail:

$$\begin{aligned}
s'(p) &= \sigma(p) - \sum_{\ell \in N} w_\ell \alpha_{v_\ell(p)} \tilde{y}_\ell + \alpha_0 W_{M'}\\
&\geq \sigma(p) - \sum_{\ell \in N} w_\ell \alpha_{v_\ell(p)} \tilde{y}_\ell + \alpha_0 W^\star - \alpha_0 R_1(\lambda, w_{\max}, W^\star) && \text{(by Lemma 8.2)}\\
&\geq T^\star - 2\alpha_0 R_1(\lambda, w_{\max}, W^\star) && \text{(by Lemma 8.4)}\\
&\geq \tilde{T} - (2\alpha_0 R_1(\lambda, w_{\max}, W^\star) + R_3) && \text{(by Lemma 11)}\\
&= \tilde{T} - R_4 \ .
\end{aligned}$$

The failure probability is bounded by $6\mathcal{N}^{-\lambda+1}$, by a union-bound on the failure probabilities of the lemmas used. □

To summarize, Lemma 12 shows that currently p might be still losing, albeit by a margin of at most R_4.

4.6.1 Bounding the Current Margin

Hitherto we have defined a series of bounds R_1, R_2, R_3, R_4. In this section we very loosely bound each of them. Let $\bar{W} = \sqrt{w_{\max} \max\{w_{\max}, W^\star\}}$. Then:

Lemma 13. *It holds that* $R_4 \leq 56\lambda^2\alpha_0\bar{W}\ln^2\mathcal{N}$.

Proof. As follows:

Bounding R_1. By its definition, it holds that $R_1(\lambda, w_{\max}, W^\star) = 6\lambda\bar{W}\ln\mathcal{N}$ and that $R_1(\lambda, \alpha_0 w_{\max}, \alpha_0 W^\star) \leq 6\lambda\alpha_0\bar{W}\ln\mathcal{N}$.

Bounding R_2. $R_2 = \alpha_0 R_1(\lambda, w_{\max}, W^\star) + \alpha_1 R_1(\lambda, w_{\max}, W^\star) \leq 2\alpha_0 R_1(\lambda, w_{\max}, W^\star) \leq 12\lambda\alpha_0\bar{W}\ln\mathcal{N}$.

Bounding R_3. For R_3:

$$\begin{aligned}
R_3 &= R_2 + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1(W^\star + R_1(\lambda, w_{\max}, W^\star))) \\
&\leq R_2 + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 W^\star) \\
&\quad + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 R_1(\lambda, w_{\max}, W^\star)) && \text{(By sub-additivity of sqrt)} \\
&\leq R_2 + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 W^\star) \\
&\quad + R_1(\lambda, \alpha_1 w_{\max}, \alpha_1 6\lambda\bar{W}\ln\mathcal{N}) \\
&\leq R_2 + 6\lambda\alpha_1\bar{W}\ln\mathcal{N} + 36\lambda^2\alpha_1\bar{W}\ln^2\mathcal{N} \\
&\leq (12 + 6 + 36)\lambda^2\alpha_0\bar{W}\ln^2\mathcal{N} \\
&\leq 54\lambda^2\alpha_0\bar{W}\ln^2\mathcal{N}
\end{aligned}$$

Bounding R_4. $R_4 = 2\alpha_0 R_1(\lambda, w_{\max}, W^\star) + R_3 \leq 56\lambda^2\alpha_0\bar{W}\ln^2\mathcal{N}$.

□

In the next sections we will show how to close this R_4 margin, and analyze the additional price required in order to do so and make p win.

Let $R_5 = \alpha_0\bar{W}\ln^3\mathcal{N}$. In particular, notice that $\alpha_0 w_{\max} = O(R_5/\ln^3\mathcal{N})$, and therefore

$$R_1(\lambda, \alpha_0 w_{\max}, R_5) = \sqrt{\alpha_0 w_{\max} R_5}\ln\mathcal{N} = O(R_5/\ln^{1/2}\mathcal{N}) \ , \tag{17}$$

and in addition

$$R_4 = O(R_5/\ln\mathcal{N}) \tag{18}$$

Using these asymptotic upper bounds, we derive:

Claim 14. *Let $\delta < 1/2$ and d be positive constants. For a large enough $\mathcal{N}$, it holds that $R_1(\lambda, \alpha_0 w_{\max}, dR_5) \leq \delta R_5$, $R_1(\lambda, w_{\max}, dR_5/\alpha_0) \leq \delta R_5/\alpha_0$ and $R_4 \leq \delta R_5$.*

Proof. This follows from eqs. (17) and (18), being asymptotic bounds. □

4.6.2 Constant Scoring Rules

We first describe how to close the gap for constant scoring rules, as will be detailed by Algorithm 3. Let us first recall some notation and define some new one. $R_4 = O(\alpha_0 \bar{W} \ln^2 \mathcal{N})$ is the bound on the margin between the current maximum competitor score and p's score. $R_5 = \alpha_0 \bar{W} \ln^3 \mathcal{N}$ is a quantity we shall immediately use, where $\bar{W} = \sqrt{w_{\max} \max\{w_{\max}, W^\star\}}$. $\mathsf{C}' = \mathsf{C} \setminus \{p\}$ is the competitor set, $s'(c)$ is the current score of c (i.e., before running Algorithm 3), and $s''(c)$ will be the score of c following Algorithm 3. For any subset $U \subseteq N$, recall that W_U is the sum of the weights of the voters in U, and that Ψ_U is the sum of the prices of the voters in U. Similarly to $s'(c)$ and $s''(c)$, we define $s'_U(c)$ to be the current score of c when taking into consideration only the votes from U. $s''_U(c)$ is defined analogously: the score of c following Algorithm 3, when taking into consideration only the votes from U. Let $r_{\max}$ denote $\max_{\ell \in N} \psi_\ell / w_\ell$, i.e., the maximum ratio between a voter's price and his weight, and likewise $\bar{r}_{\max} = \max_{\ell \in N} w_\ell / \psi_\ell$ denote the maximum ratio between a voter's weight and his price. Notice that it always holds that $\bar{r}_{\max} r_{\max} \geq 1$.

Let us overview Algorithm 3. We let U be the subset of voters who gave p a score of at most $\alpha_0 - 1$ (so, for instance, if for our rule $\mathcal{R}_{\boldsymbol{\alpha}}$ it holds that $\alpha_1 = \alpha_0$, then U will not include voters who awarded p an α_1-score). Given U, we shall choose a subset $M'' \subseteq U$ of additional voters to bribe, as follows. If $W_U < 2R_5/\alpha_0$, we choose $M'' = U$. Otherwise, we initialize $M'' = \emptyset$ and start adding voters from U to M'' by some arbitrary order until $W_{M''} \geq R_5/\alpha_0$. Once we have finalized M'', it is sufficient to bribe each voter in M'', where each newly bribed voter will move p to the top position, without changing the rest of his preference order. In Lemmas 15 and 16 we will prove that p is winning at the end of this procedure.

Algorithm 3: Closing the gap for constant scoring rules

```
1 Let U be the subset of voters who gave p a score of at most α_0 − 1.
2 if W_U ≤ 2R_5/α_0 then
3 |   M'' ← U
4 else
5 |   M'' ← ∅
6 |   foreach ℓ ∈ U do
7 |   |   M'' ← M'' ∪ {ℓ}
8 |   |   if W_{M''} ≥ R_5/α_0 then break
9 For each voter ℓ in M'', bribe him and change his preference order such that p is
    moved to the top position, without any further changes to his preference order.
```

Lemma 15. *In Algorithm 3, it holds that $W_{M''} \leq 2R_5/\alpha_0$ and $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$. In addition, if $M'' \subset U$, then $W_{M''} \geq R_5/\alpha_0$ as well.*

Proof. Observe the condition of the 'if' statement in Line 2 of Algorithm 3. If it is true then $W_{M''} \leq 2R_5/\alpha_0$ and therefore $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$ by the definition of $r_{\max}$.

Otherwise M'' was built by iteratively adding voters. Let ℓ be the last voter added to M''. Then $W_{M''} = W_{M'' \setminus \{\ell\}} + w_\ell \leq R_5/\alpha_0 + w_{\max}$, where the last inequality holds since

before adding ℓ, $W_{M''\setminus\{\ell\}}$ was at most R_5/α_0. In addition, since also $w_{\max} \leq R_5/\alpha_0$ (by the definition of R_5), we obtain that $W_{M''} \leq 2R_5/\alpha_0$. As for the cost, $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$ by the definition of $r_{\max}$.

In the specific case where $M'' \subset U$, we are certain that the condition of the 'if' statement in Line 2 of Algorithm 3 was false. Thus, $W_{M''} \geq R_5/\alpha_0$ by Line 8 of the algorithm. □

In the next lemma we show that for constant voting rules, Algorithm 3 closes the margin by which p is possibly losing.

Lemma 16. *Let $\mathcal{R}_{\boldsymbol{\alpha}}$ be a constant scoring rule. Assuming all previous lemmas had succeeded, then for a large enough $\mathcal{N}$, Algorithm 3 makes p win by paying an additional cost of at most $2R_5 r_{\max}/\alpha_0$.*

Proof. Recall that U is the set of voters who gave p a score of at most $\alpha_0 - 1$. Obviously, all of them are un-bribed voters, since, when the algorithm bribes a voter, p is awarded the top score α_0 by this voter. Given U, we have built the subset $M'' \subseteq U$ of additional voters to be bribed. Recall that such a subset has a price of at most $2R_5 r_{\max}/\alpha_0$. Recall that we bribed each voter in M'', such that each newly bribed voter moved p to the top position, without changing the rest of his preference order. We now split our proof into cases:

- If $M'' \subset U$, let I be the *additional* points awarded to p as a result of this new bribery. It holds that $I = \sum_{\ell \in M''} (\alpha_0 - \alpha_{v_\ell(p)}) w_\ell$. In particular notice that $I \geq I'$ where $I' = \sum_{\ell \in M''} w_\ell = W_{M''} \geq R_5/\alpha_0$, since every bribed voter in M'' now awards p at least one extra point. Then:

$$\begin{aligned} I &\geq I' \\ &\geq R_5/\alpha_0 \\ &\geq R_4 \qquad\qquad \text{(by Claim 14 with } \delta = 1/(2\alpha_0)) \end{aligned}$$

  where the last inequality holds asymptotically (otherwise $\mathcal{N}$ is constant and we can solve the problem in constant time).

- In the 'degenerate' case where $M'' = U$, we bribed all voters in U, and described a scenario where every voter was either bribed or gave p a score equal to α_0 in the first place. According to the co-winner assumption, this is sufficient to make p a winner.

In conclusion, in both cases we covered the R_4 gap, thus p now wins. The overall price paid is $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$ by Lemma 15. □

4.6.3 Non-Concentrated Scoring Rules

The reasoning we have used for constant scoring rules does not necessarily apply to non-concentrated scoring rules. To see why, notice that for constant scoring rules—as demonstrated by the previous lemma—a fairly simple 'constructive' argument was sufficient: it was enough for us to analyze how many points were contributed to p's score by each newly bribed voter ℓ, obviating the need to also analyze the points lost by her competitors. The main observation there was that p receives $w_\ell \cdot \Theta(\alpha_0)$ additional points from any newly bribed voter ℓ – a relatively large number. This was guaranteed by the fact that we only

bribed voters who previously awarded p a score α_j for which $\alpha_j < \alpha_0$, and thus, as the scores are integral, the contribution in points to p's score was $(\alpha_0 - \alpha_j)w_\ell \geq w_\ell = w_\ell \cdot \Theta(\alpha_0)$.

Non-concentrated rules $\mathcal{R}_{\boldsymbol{\alpha}}$ will have a much more involved proof in which it is often insufficient to analyze only the direct contribution to p's score by a newly bribed voter; we also have to analyze the indirect contribution to p's score incurred by p's competitors losing points.

In the following, we shall assume w.l.o.g. that for a non-concentrated scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, it holds that $\bar{\alpha} \leq (1 - 5\epsilon)\alpha_0$ for some constant $0 < \epsilon < 1/5$. Our method is described as Algorithm 4. Before going over its main ideas, we recall the notation involved: $R_4 = O(\alpha_0 \bar{W} \ln^2 \mathcal{N})$ is the bound on the margin between the current maximum competitor score and p's score. $R_5 = \alpha_0 \bar{W} \ln^3 \mathcal{N}$, where $\bar{W} = \sqrt{w_{\max} \max\{w_{\max}, W^\star\}}$. $\mathsf{C}' = \mathsf{C} \setminus \{p\}$ is the competitor set, N is the voter set, and $M' \subseteq N$ is the set of voters we have bribed thus far. For any subset $U \subseteq N$, $W_U = \sum_{\ell \in U} w_\ell$ and $\Psi_U = \sum_{\ell \in U} \psi_\ell$ are the aggregate weight and price, respectively, of the voters in U. $s'_U(c)$ is the score of c before running Algorithm 4, when taking into consideration only the votes from U. $s''_U(c)$ is the score of c after running Algorithm 4, again when taking into consideration only the votes from U. $s'(c)$ is a shorthand for $s'_N(c)$, and $s''(c)$ for $s''_N(c)$. As before, $r_{\max} = \max_{\ell \in N} \psi_\ell / w_\ell$ and $\bar{r}_{\max} = \max_{\ell \in N} w_\ell / \psi_\ell$. As mentioned, it always holds that $\bar{r}_{\max} r_{\max} \geq 1$.

Let us briefly go over Algorithm 4's main ideas. In its first part, we need to choose which voters to bribe (denoted as the set M''). To do this, we observe the set $U = N \setminus M'$ of currently un-bribed voters. If its overall weight is small (Line 2), we simply set $M'' \leftarrow U$. Otherwise, we randomly choose voters for participation in M''. Once we have finalized M'', we bribe these voters and need to decide on their new ballots. Here there are two scenarios: if p's current score is relatively small (Line 8), it is sufficient to change their ballots such that p is promoted to the top position in their preference orders, without any further changes. Otherwise, we replace each of their ballots with a new one in which p is in the top position and all other candidates are ranked randomly. However, the way we describe this is slightly peculiar: we first draw a potential ballot v'_ℓ (with p at the top position, and all others ranked randomly, as described) for *every* voter in U (even the ones not chosen for M'') without actually assigning this potential ballot to the voter. Only then we go over the voters in M'', where each of them will be assigned his respective potential ballot. The potential ballots we prepared for voters in $U \setminus M''$ are effectively discarded and are never used.

Obviously, it is sufficient and more efficient to compute the new, potential ballots v'_ℓ only for the voters in M''. However, presenting the algorithm in this way helps the argument in Lemma 18. Specifically, the point is to emphasize that the two random decisions pertaining to each voter $\ell \in U$, of (a) whether ℓ is chosen to be bribed, and (b) the value of his *potential* ballot v'_ℓ, are independent of one another.

We shall require the following lemmas, in which we analyze the algorithm by splitting it into cases. In the first lemma we study the scenario where the condition in Line 2 of Algorithm 4 is true.

Lemma 17. *Assuming all previous lemmas had succeeded, if the condition in Line 2 of Algorithm 4 is true, then Algorithm 4 makes p win by paying an additional cost of at most $2R_5 r_{\max}/\alpha_0$.*

Algorithm 4: Closing the gap for non-concentrated scoring rules

```
   /* Choosing the set M'' of additional voters to bribe:                    */
 1 Let U = N \ M' be the set of un-bribed voters.
 2 if W_U < 2R_5/α_0 then
 3 |  M'' ← U
 4 else
 5 |  M'' ← ∅
 6 |  foreach ℓ ∈ U do
 7 |  |  With probability R_5/(α_0 W_U): set M'' ← M'' ∪ {ℓ}.
   /* Calculate the strategy for the voters in M'':                          */
 8 if M'' = U or s'_U(p) ≤ (1 − ε)α_0 W_U then
 9 |  For each voter ℓ in M'', bribe him and change his preference order such that p is
   |    moved to the top position (unless p is already ranked at the top), without any
   |    further changes to his preference order.
10 else
   |  /* Prepare a potential ballot for each voter in U:                     */
11 |  For each voter ℓ in U, define a potential ballot v'_ℓ as follows: in v'_ℓ, p is moved to
   |    the top position, and all other candidates are ranked randomly in the remaining
   |    positions, that is, the ranking of all other candidates will be determined by a
   |    random permutation.
   |  /* Assign the potential ballots only to the voters in M'':            */
12 |  For each voter ℓ in M'', bribe him and set v_ℓ ← v'_ℓ (such that ℓ now awards α_{v'_ℓ(c)}
   |    to candidate c).
```

Proof. Having the condition in Line 2 of Algorithm 4 being true leads to a 'degenerate' case where $M'' = U$. In this case we reach Line 9, and we bribe all voters in U. Since all voters in $N \setminus U$ are already bribed as well, we reach a scenario where every voter is bribed. Since every bribed voter ranks p at the top, then according to the co-winner assumption, this suffices to make p a winner.

Since by the condition in Line 2 of Algorithm 4 being true it holds that $W_{M''} \leq 2R_5/\alpha_0$, we get that $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$ by the definition of $r_{\max}$. □

Define ratios $\rho_c \in [0,1]$ for each $c \in \mathsf{C}$ such that $s'_U(c) = \rho_c \alpha_0 W_U$. In words, the ρ_c value is the ratio between c's score from the voters in U (i.e., $s'_U(c)$) and the maximum score obtainable by a candidate given only the voters in U (which is $\alpha_0 W_U$). We will now study the case where the condition in Line 2 of Algorithm 4 is false.

Lemma 18. *Assuming that all previous lemmas had succeeded, that the condition in Line 2 of Algorithm 4 is false, and that $\mathcal{N}$ is large enough, then with a failure probability of at most $3\mathcal{N}^{-\lambda+1}$ all the following hold:*

1. $W_{M''} \in [R_5/\alpha_0 \pm \epsilon/10 \cdot R_5/\alpha_0]$ *and* $R_5/\alpha_0 \leq (1 + 2\epsilon/10) W_{M''}$.
2. $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$.
3. $s''_{M''}(p) \geq R_5 - \epsilon/10 \cdot R_5$.

4. *For each* $c \in \mathsf{C}$*:* $s'_{M''}(c) \in [\rho_c R_5 \pm \epsilon/10 \cdot R_5]$.

5. *If Algorithm 4 reached Line 11 then for each* $c \in \mathsf{C}'$*:* $s''_{M''}(c) \in [\bar{\alpha} R_5/\alpha_0 \pm \epsilon/10 \cdot R_5]$.

Proof. By the condition in Line 2 of Algorithm 4 being false, M'' was built by randomly adding voters.

1. $\mathbb{E}[W_{M''}] = \sum_{\ell \in U} R_5/(\alpha_0 W_U) \cdot w_\ell = R_5/\alpha_0$. According to Lemma 1, $W_{M''} \in [R_5/\alpha_0 \pm R_1(\lambda, w_{\max}, R_5/\alpha_0)]$. Using Claim 14 and choosing $\delta = \epsilon/10$, we obtain that $W_{M''} \in [R_5/\alpha_0 \pm \epsilon/10 \cdot R_5/\alpha_0]$. Simple arithmetic shows that if $W_{M''} \geq R_5/\alpha_0 - \epsilon/10 \cdot R_5/\alpha_0$, then $R_5/\alpha_0 \leq (1 + 2\epsilon/10) W_{M''}$.

2. It follows from the previous item that $\Psi_{M''} \leq (1 + \epsilon/10) R_5 r_{\max}/\alpha_0 \leq 2R_5 r_{\max}/\alpha_0$ (by the definition of $r_{\max}$).

3. It follows from the first item that $s''_{M''}(p) = \alpha_0 W_{M''} \geq R_5 - \epsilon/10 \cdot R_5$.

4. For any candidate $c \in \mathsf{C}$, notice that $\mathbb{E}[s'_{M''}(c)] = \sum_{\ell \in U} R_5/(\alpha_0 W_U) \cdot w_\ell \alpha_{v_\ell(c)} = \rho_c R_5$. According to Lemma 1, $s'_{M''}(c) \in [\rho_c R_5 \pm R_1(\lambda, \alpha_0 w_{\max}, \rho_c R_5)]$. Following Claim 14 with $\delta = \epsilon/10$ we obtain that $s'_{M''}(c) \in [\rho_c R_5 \pm \epsilon/10 \cdot R_5]$.

5. In the case Algorithm 4 reached Line 11, we can show that $s''_{M''}(c) \in [\bar{\alpha} R_5/\alpha_0 \pm \epsilon/10 \cdot R_5]$ as follows: by definition, $s''_{M''}(c) = \sum_{\ell \in U} \mathbb{1}_{M''}(\ell) w_\ell \alpha_{v'_\ell(c)}$, where $\mathbb{1}_{M''}(\ell)$ is an indicator function for ℓ's membership in M'' and v'_ℓ for each ℓ are the new potential ballots. Then

$$\begin{aligned}
\mathbb{E}[s''_{M''}(c)] &= \sum_{\ell \in U} w_\ell \mathbb{E}[\mathbb{1}_{M''}(\ell) \alpha_{v'_\ell(c)}] \\
&= \sum_{\ell \in U} w_\ell \mathbb{E}[\mathbb{1}_{M''}(\ell)] \cdot \mathbb{E}[\alpha_{v'_\ell(c)}] \quad (\mathbb{1}_{M''}(\ell),\ \alpha_{v'_\ell(c)} \text{ are mutually independent}) \\
&= \sum_{\ell \in U} w_\ell R_5/(\alpha_0 W_U) \cdot \bar{\alpha} \\
&= W_U R_5/(\alpha_0 W_U) \cdot \bar{\alpha} \\
&= \bar{\alpha} R_5/\alpha_0 \ .
\end{aligned}$$

   The second equality follows from $\mathbb{1}_{M''}(\ell)$ and $\alpha_{v'_\ell(c)}$ being independent of one another, as discussed in our overview of Algorithm 4. From this point, we use Lemma 1 and the same reasoning for $s''_{M''}(c)$, as we have just done for $s'_{M''}(c)$.

The overall failure probability follows by a union-bound over the required applications of Lemma 1, once for Item 1, and at most $m+1$ times (for each candidate) for each of Items 4 and 5. □

The next lemma studies the case where $M'' \subset U$ and $s'_U(p) \leq (1 - \epsilon)\alpha_0 W_U$, and thus the algorithm reaches Line 9.

Lemma 19. *Assuming all previous lemmas had succeeded, if* M'' *was built by randomly adding voters, and* $s'_U(p) \leq (1 - \epsilon)\alpha_0 W_U$*, then for a large enough* $\mathcal{N}$*, Algorithm 4 makes* p *win by paying an additional cost of at most* $2R_5 r_{\max}/\alpha_0$.

Proof. Since $s'_U(p) \leq (1-\epsilon)\alpha_0 W_U$, then according to Lemma 18, $s'_{M''}(p) \leq (1-\epsilon)R_5 + \epsilon/10 \cdot R_5$ and in addition $s''_{M''}(p) = \alpha_0 W_{M''} \geq R_5 - \epsilon/10 \cdot R_5$. In this case each newly bribed voter moves p to the top position (unless p is already ranked at the top), without any other changes. Notice that the score of any of p's competitors can only decrease as a result of such a change. Let I be the increase in p's score. Then:

$$\begin{aligned} I &= s''_N(p) - s'_N(p) \\ &= s''_{M''}(p) - s'_{M''}(p) && \text{(since } s''_{N\setminus M''}(p) = s'_{N\setminus M''}(p)\text{)} \\ &\geq R_5 - (1-\epsilon)R_5 - 2\epsilon/10 \cdot R_5 \\ &= 8\epsilon/10 \cdot R_5 \\ &> R_4 && \text{(by Claim 14 with } \delta = 8\epsilon/10\text{)} \end{aligned}$$

where the last inequality holds for a large enough $\mathcal{N}$.

In summary we covered the R_4 gap, thus p now wins. The overall price paid is $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$ by Lemma 18. □

The next lemma studies the case where Algorithm 4 reaches Lines 11 and 12.

Lemma 20. *Assuming all previous lemmas had succeeded, if M'' was built by randomly adding voters, and $s'_U(p) > (1-\epsilon)\alpha_0 W_U$, then for a large enough $\mathcal{N}$, Algorithm 4 makes p win by paying an additional cost of at most $2R_5 r_{\max}/\alpha_0$.*

Proof. In this case, p was already awarded a high score from the voters in U. The algorithm, for each newly bribed voter, moves p to the top position, and in addition ranks all other candidates according to a random permutation. This time we shall focus on the competitors. Fix some competitor $c \in \mathsf{C}'$, and let I_c be the (possibly negative) increase in c's score.

In Lemma 18, we showed that for each $c \in \mathsf{C}'$, $s''_{M''}(c) \leq \bar{\alpha} R_5/\alpha_0 + \epsilon/10 \cdot R_5$. Therefore:

$$\begin{aligned} I_c &= s''_N(c) - s'_N(c) \\ &= s''_{M''}(c) - s'_{M''}(c) && \text{(since } s''_{N\setminus M''}(p) = s'_{N\setminus M''}(c)\text{)} \\ &\leq \bar{\alpha} R_5/\alpha_0 - s'_{M''}(c) + \epsilon/10 \cdot R_5 && \text{(by Lemma 18)} \end{aligned}$$

We now split our proof into cases according to $s'_U(c)$:

- c is a highly-ranked candidate, such that $s'_U(c) \geq (1-3\epsilon)\alpha_0 W_U$. By Lemma 18 with $\rho_c \geq 1-3\epsilon$, it holds that $s'_{M''}(c) \geq (1-3\epsilon)R_5 - \epsilon/10 \cdot R_5$. Therefore:

  $$\begin{aligned} I_c &\leq \bar{\alpha} R_5/\alpha_0 - s'_{M''}(c) + \epsilon/10 \cdot R_5 \\ &\leq \bar{\alpha} R_5/\alpha_0 - (1-3\epsilon)R_5 + 2\epsilon/10 \cdot R_5 \\ &\leq (1-5\epsilon)R_5 - (1-3\epsilon)R_5 + 2\epsilon/10 \cdot R_5 \\ &< -\epsilon R_5 \\ &\leq -R_4 && \text{(by Claim 14 with } \delta = \epsilon\text{)} \end{aligned}$$

  where the last inequality holds for a large enough $\mathcal{N}$. Candidate c has just lost more than R_4 points. Since the gap between any candidate c and p was at most R_4, and p can only gain points by the bribery, c now has fewer points compared to p.

- c is a slightly above average candidate, such that $\bar{\alpha}W_U \leq s'_U(c) \leq (1-3\epsilon)\alpha_0 W_U$. By Lemma 18 with $\bar{\alpha}/\alpha_0 \leq \rho_c \leq 1-3\epsilon$, it holds that $(\bar{\alpha}/\alpha_0)R_5 - \epsilon/10 \cdot R_5 \leq s'_{M''}(c) \leq (1-3\epsilon)R_5 + \epsilon/10 \cdot R_5$. Therefore:

$$\begin{aligned} I_c &\leq \bar{\alpha}R_5/\alpha_0 - s'_{M''}(c) + \epsilon/10 \cdot R_5 \\ &\leq \bar{\alpha}R_5/\alpha_0 - (\bar{\alpha}/\alpha_0)R_5 + 2\epsilon/10 \cdot R_5 \\ &\leq 0 + 2\epsilon/10 \cdot R_5 \\ &\leq 2\epsilon/10 \cdot (1+2\epsilon/10)\alpha_0 W_{M''} && \text{(by Lemma 18)} \\ &\leq \epsilon\alpha_0 W_{M''} && \text{(by simple arithmetic)} \end{aligned}$$

  where the one before last inequality holds for a large enough $\mathcal{N}$.

  To summarize this case, c's score might have increased, but only by at most $\epsilon\alpha_0 W_{M''}$ and thus $s''_U(c) \leq (1-3\epsilon)\alpha_0 W_U + \epsilon\alpha_0 W_{M''} \leq (1-2\epsilon)\alpha_0 W_U < s'_U(p) \leq s''_U(p)$, so p has a higher score than c given the voters in U. The rest of the voters—those in $N \setminus U$—are already bribed, thus $s''_{N\setminus U}(p) \geq s''_{N\setminus U}(c)$, and we get that $s''_N(p) \geq s''_N(c)$, i.e., c now has fewer points compared to p.

- c is a below average candidate, such that $s'_U(c) \leq \bar{\alpha}W_U$ and thus $\rho_c \leq \bar{\alpha}/\alpha_0$. By Lemma 18, $s'_U(c) = \rho_c\alpha_0 W_U$ implies that $s'_{M''}(c) \geq \rho_c R_5 - \epsilon/10 \cdot R_5$. Therefore:

$$\begin{aligned} I_c &\leq \bar{\alpha}R_5/\alpha_0 - s'_{M''}(c) + \epsilon/10 \cdot R_5 \\ &= (\bar{\alpha}/\alpha_0 - \rho_c + 2\epsilon/10)R_5 \\ &\leq (\bar{\alpha}/\alpha_0 - \rho_c + 2\epsilon/10) \cdot (1+2\epsilon/10)\alpha_0 W_{M''} && \text{(by Lemma 18)} \\ &\leq (\bar{\alpha}/\alpha_0 - \rho_c + 2\epsilon/10) \cdot (1+2\epsilon/10)\alpha_0 W_U && \text{(since } \bar{\alpha}/\alpha_0 - \rho_c + 2\epsilon/10 > 0 \\ & && \text{and } W_{M''} \leq W_U) \\ &\leq \bar{\alpha}W_U - \rho_c\alpha_0 W_U + \epsilon\alpha_0 W_U && \text{(by simple arithmetic and } \bar{\alpha}/\alpha_0 < 1) \end{aligned}$$

  Moving to $s''_U(c)$, then:

$$\begin{aligned} s''_U(c) &= s'_U(c) + I_c \\ &\leq \rho_c\alpha_0 W_U + (\bar{\alpha}W_U - \rho_c\alpha_0 W_U + \epsilon\alpha_0 W_U) \\ &\leq \bar{\alpha}W_U + \epsilon\alpha_0 W_U \\ &\leq (1-4\epsilon)\alpha_0 W_U \\ &< s'_U(p) \\ &\leq s''_U(p) \end{aligned}$$

  To summarize this case, c's score might have increased, but p still has a higher score than c given the voters in U. The rest of the voters—those in $N \setminus U$—are already bribed, thus $s''_{N\setminus U}(p) \geq s''_{N\setminus U}(c)$, and we get that $s''_N(p) \geq s''_N(c)$, i.e., c now has fewer points compared to p.

In conclusion, we covered any positive gap between any voter and p, thus p now wins. The overall price paid is $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$ by Lemma 18. □

In the following, we return to discussing constant and non-concentrated scoring rules jointly. With Lemmas 15 to 20, we showed that for many types of $\boldsymbol{\alpha}$, we can close the margin while paying an additional cost of at most $2R_5 r_{\max}/\alpha_0$, with a high probability. Recall the value $\Psi^\star$ in the solution of LP_{B}; its counterpart $\Psi_{M'}$ which is the sum of the prices of the voters in M' (the initial set of voters which we bribed); and $\psi_{\max}$, the maximum voter price. Let $\bar{\psi} = \sqrt{\psi_{\max} \max\{\psi_{\max}, \Psi^\star\}}$. This leads to the following.

Lemma 21. *In SR-*Weighted-$Bribery *under constant or non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, assuming that Lemma 12 did not fail, then for a large enough $\mathcal{N}$, besides the $\Psi_{M'}$ cost we already paid, with a failure probability of at most $3\mathcal{N}^{-\lambda+1}$, an additional cost of at most $\Psi_{M''} = \widetilde{O}(\bar{\psi}\bar{r}_{\max} r_{\max})$ is needed for p to win.*

Proof. According to either Lemmas 15 and 16 (constant scoring rules), or Lemmas 17 to 20 (non-concentrated scoring rules), it holds that $\Psi_{M''} \leq 2R_5 r_{\max}/\alpha_0$. Recall that $\bar{W} = \sqrt{w_{\max} \max\{w_{\max}, W^\star\}}$ and $R_5 = \alpha_0 \bar{W} \ln^3 \mathcal{N}$. Since $W^\star \leq \Psi^\star \bar{r}_{\max}$ and $w_{\max} \leq \psi_{\max} \bar{r}_{\max}$, then $R_5 = \alpha_0 \bar{\psi} \bar{r}_{\max} \ln^3 \mathcal{N}$. Plugging this into $\Psi_{M''} = 2R_5 r_{\max}/\alpha_0$, we get that $\Psi_{M''} = \widetilde{O}(\bar{\psi}\bar{r}_{\max} r_{\max})$. The failure probability follows from the failure probability of Lemma 18 (for non-concentrated scoring rules; for constant scoring rules notice that there are no randomized actions executed by Algorithm 3). □

We are now ready for the main theorem of this section:

Theorem 22. *In SR-*Weighted-$Bribery *under constant or non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, let Ψ^{OPT} be the minimum cost of bribery in order to make p win. Then there exists a polynomial-time randomized algorithm spending a cost of at most*

$$\Psi^{\mathsf{OPT}} + \widetilde{O}\left(\sqrt{\psi_{\max}(\Psi^{\mathsf{OPT}} + \psi_{\max})}\bar{r}_{\max} r_{\max}\right) ,$$

with an exponentially-small failure probability.

Proof. Ψ^{OPT} is the cost of bribery according to an optimal strategy, and thus $\Psi^\star \leq \Psi^{\mathsf{OPT}}$ since the integral optimum cannot be smaller than the fractional one. Using Lemma 8.1, overall we spent $\Psi_{M'} + \Psi_{M''} \leq \Psi^\star + R_1(\lambda, \psi_{\max}, \Psi^\star) + \Psi_{M''} \leq \Psi^{\mathsf{OPT}} + R_1(\lambda, \psi_{\max}, \Psi^\star) + \Psi_{M''}$.

For the first additive term $R_1(\lambda, \psi_{\max}, \Psi^\star)$, notice that

$$R_1(\lambda, \psi_{\max}, \Psi^\star) = O\left(\sqrt{\psi_{\max} \max\{\psi_{\max}, \Psi^{\mathsf{OPT}}\}} \ln \mathcal{N}\right) .$$

For the second additive term $\Psi_{M''}$, consider Lemma 21; we can always assume w.l.o.g. that $\mathcal{N}$ is large enough for the lemma (and the lemmas it depends upon) to hold—as required by it—otherwise $\mathcal{N} = n(m+1)$ is constant and the entire problem can be easily solved. Thus, according to Lemma 21,

$$\begin{aligned} \Psi_{M''} &= \widetilde{O}(\bar{\psi}\bar{r}_{\max} r_{\max}) \\ &= \widetilde{O}\left(\sqrt{\psi_{\max} \max\{\psi_{\max}, \Psi^{\mathsf{OPT}}\}}\bar{r}_{\max} r_{\max}\right) . \end{aligned}$$

Combining both terms, and recalling that $\bar{r}_{\max} r_{\max} \geq 1$, overall we get that

$$\Psi_{M'} + \Psi_{M''} \leq \Psi^{\mathsf{OPT}} + \widetilde{O}\left(\sqrt{\psi_{\max}(\psi_{\max} + \Psi^{\mathsf{OPT}})}\bar{r}_{\max} r_{\max}\right) .$$

The failure probability is at most $10\mathcal{N}^{-\lambda+1}$ by a union bound over the failure probabilities of Lemma 8.1, Lemma 12, and Lemma 21. By choosing e.g., $\lambda = 2$, the failure probability is bounded by $10\mathcal{N}^{-1} = 1/\Omega(\mathcal{N})$. By running the algorithm a linear number of times, and choosing the run yielding the minimal overall cost, the failure probability becomes exponentially-small, while the runtime stays polynomial. □

This theorem immediately yields the two following interesting cases.

Corollary 23. *In SR-*Weighted-$Bribery *under constant or non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, let Ψ^{OPT} be the minimum cost of bribery in order to make p win. If voter weights and voter prices are constant in the input size, then there exists a polynomial-time randomized algorithm spending a cost of at most*

$$\Psi^{\mathsf{OPT}} + \widetilde{O}\left(\sqrt{\Psi^{\mathsf{OPT}}}\right) ,$$

with an exponentially-small failure probability.

Proof. By $\psi_{\max}$ and $\bar{r}_{\max} r_{\max}$ being constant when both prices and weights are constant. □

Remark. Recall that the $\widetilde{O}(\sqrt{\Psi^{\mathsf{OPT}}})$ notation hides terms which are poly-logarithmic in $\mathcal{N}$. Therefore, the approximation described by Corollary 23 does not qualify as an asymptotic polynomial-time approximation scheme.[4]

Corollary 24. *In SR-*Weighted-Bribery *under constant or non-concentrated $\mathcal{R}_{\boldsymbol{\alpha}}$, let k^{OPT} be the minimum number of voters to be bribed in order to make p win. Then there exists a polynomial-time randomized algorithm bribing at most $k^{\mathsf{OPT}} + \widetilde{O}(\sqrt{k^{\mathsf{OPT}}} w_{\max}/w_{\min})$ voters, with an exponentially-small failure probability, where $w_{\max}$ (resp. $w_{\min}$) is the maximum (resp. minimum) voter weight.*

Proof. For unit-priced voters, $\Psi^{\mathsf{OPT}} = k^{\mathsf{OPT}}$ and $\bar{r}_{\max} r_{\max} = w_{\max}/w_{\min}$. □

5. Related Work

In this section we detail some of the previous work to help present our new results in context. We aim at maintaining the introductory style of this section; for formal definitions, see Section 2.

Remark. When discussing results that pertain to any scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$—and not only to a specific one—notice that most previous work had made the simplifying assumption that the number of candidates $|\mathsf{C}|$ has to be fixed, as a result of $\boldsymbol{\alpha}$ being hard-coded into the algorithm. See further details below.

4. An *asymptotic polynomial-time approximation scheme* (Asymptotic PTAS) is an approximation scheme in which for any constant $\epsilon > 0$, the scheme can yield an approximation algorithm where the additive approximation factor is bounded by $\epsilon\Psi^{\mathsf{OPT}} + f(\epsilon)$, where $f(\epsilon)$ in a term that depends only on ϵ.

5.1 Coalitional Manipulation

Tractability Results. The computational complexity of coalitional manipulation problems has been studied extensively. For any scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, much of the earlier work considered the case where the number of candidates is bounded: Conitzer, Sandholm, and Lang (2007, see Proposition 1) show that when $|\mathsf{C}|$ is bounded, $\mathcal{R}_{\boldsymbol{\alpha}}$-UCM is solvable in polynomial time.

Even when $|\mathsf{C}|$ is unbounded, Plurality-UCM and Veto-UCM are still easy, and can be solved using Reverse, Zuckerman et al.'s (2009) greedy algorithm.[5] More generally, t-approval-UCM is easy as well (Xia et al., 2010).

Recently it was shown by Hemaspaandra and Schnoor (2016) that for every scoring rule in which $\boldsymbol{\alpha}$ consists of a constant number of unique coefficients, UCM is easy as well.

NP-Hardness Results. In the weighted case, the situation is different. For all positional scoring rules $\mathcal{R}_{\boldsymbol{\alpha}}$, except Plurality-like rules, $\mathcal{R}_{\boldsymbol{\alpha}}$-WCM is NP-hard when $|\mathsf{C}| \geq 3$ (Conitzer et al., 2007; Hemaspaandra & Hemaspaandra, 2007; Procaccia & Rosenschein, 2007). However, these results are based on a reduction from the well-known *partition* problem, which has a pseudo-polynomial algorithm; therefore, they do not extend to the case where the weights are relatively small, e.g., are integers bounded by a polynomial in the input size. When this is indeed the case, or that the weights are encoded by a unary encoding, then if $|\mathsf{C}|$ is constant, it holds that $\mathcal{R}_{\boldsymbol{\alpha}}$-WCM is easy (Faliszewski et al., 2009, with the main insight being that for a fixed $|\mathsf{C}|$, the hardness of the problem *depends* on the weights being large); if $|\mathsf{C}|$ is not fixed, then even Veto-WCM, and t-approval-WCM for $t \geq 2$ are NP-hard, by a reduction from *unary-3-partition* (Brelsford et al., 2008). The computational hardness of Borda-UCM remained open for quite some time, until it was finally shown to be NP-hard as well (Davies et al., 2014; Betzler, Niedermeier, & Woeginger, 2011), even for the case of $n = 3$ and adding $k = 2$ manipulators.

Approximating the number of manipulators. Zuckerman et al. (2009) presented a greedy algorithm later referred to as Reverse. For Borda-UCM, Reverse can be seen as an additive $+1$-approximation for the objective of finding the minimum number of manipulators needed.

For Borda-WCM, their approximation can be described as follows. Let $\mathbf{w} = (w_\ell)_{\ell \in M}$ be the weights of the k given weighted manipulators M. If a p-winning strategy using these k manipulators exists, a p-winning strategy using additional manipulators will be found if the sum of the weights of the additional manipulators equals $\max_{\ell \in M} w_\ell$.

As for more general results, Xia et al. (2010) provide an additive $(|\mathsf{C}|-2)$-approximation for SR-UCM. In the case of SR-WCM, each of the extra manipulators will have a weight of at most $\max_{\ell \in M} w_\ell/2$.

Approximating the Score Margin and the Maximum Competitor Score. Another line of work had focused on minimizing the *margin* between the highest-scoring competitor and p (the *score margin*), or some additive function thereof which also attains its minimum when the margin is minimized. Such approximations boil down to approximating the maximum score of a competitor. For $\mathcal{R}_{\boldsymbol{\alpha}}$-WCM, when $|\mathsf{C}|$ is bounded, Brelsford et al.

5. The name of the algorithm was given by Davies, Katsirelos, Narodytska, Walsh, and Xia (2014).

(2008, see Lemma 3) provide an FPTAS with respect to the maximum score of a competitor. In their work, this FPTAS paves the way for an FPTAS for another objective, namely the difference between the score margin when not including the manipulators' votes and the optimal score margin when including them. Notice that the maximum competitor score is the only nontrivial value in this computation. Keller et al. (2019) provide an additive approximation to the maximum competitor score for general values of $|\mathsf{C}|$.

5.2 Bribery

As discussed, a string of results researched both the hardness and approximability of UCM for various voting rules, and in particular scoring rules. However, it seems that the equivalent landscape for Bribery is lacking; only little work was done on approximating Bribery. We briefly cover the landscape w.r.t. Bribery, with a focus on scoring rules.

In addition to the basic, unit-price Bribery, where all voters have a unit-price for changing their ballot, other price functions exist; besides \$Bribery—discussed in this paper—in some other schemes the price changes on the basis of the operation requested by the bribing entity; in *Swap Bribery*, each voter has a price for swapping two consecutively-ranked candidates, and the price might depend in their identity. *Shift Bribery* is the same, where only swaps promoting p are allowed.

Faliszewski et al. (2009) first studied the different bribery settings w.r.t. Plurality, when $|\mathsf{C}|$ is not fixed. They show that Plurality-Weighted-\$Bribery is NP-hard for general (i.e., possibly large) weights and prices, but that both Plurality-\$Bribery and Plurality-Weighted-Bribery are easy. Trying to understand if the hardness of Plurality-Weighted-\$Bribery relies on the large values of the weights and prices, they showed that this is indeed the case: Plurality-Weighted-\$Bribery is in P if *either* weights or prices are encoded in unary.

Moving away from Plurality, the situation quickly changes: t-approval-Bribery is NP-hard (Lin, 2012) except for some small values of t for which t-approval-Bribery and t-Veto-Bribery are still easy (Faliszewski et al., 2009; Lin, 2012). Borda-Bribery is NP-hard as well (Brelsford et al., 2008).

Moving to general scoring rules, when $|\mathsf{C}|$ is constant, Faliszewski et al. (2009) show that for any scoring rule $\mathcal{R}_{\boldsymbol{\alpha}}$, $\mathcal{R}_{\boldsymbol{\alpha}}$-Weighted-\$Bribery is NP-hard. $\mathcal{R}_{\boldsymbol{\alpha}}$-Weighted-Bribery is still NP-hard for all $\mathcal{R}_{\boldsymbol{\alpha}}$, with the exception of Plurality-like rules. When limiting the weights to be unary, $\mathcal{R}_{\boldsymbol{\alpha}}$-Weighted-\$Bribery becomes easy, making this yet another example where hardness depends on the weights being large. In particular, this implies that $\mathcal{R}_{\boldsymbol{\alpha}}$-\$Bribery is easy.

We are left with the case of general scoring rules when $|\mathsf{C}|$ is not fixed, which lies at the heart of this work. This general problem—assuming that $\mathcal{R}_{\boldsymbol{\alpha}}$ is given as part of the input—is NP-hard even for its simplest variant, SR-Bribery. This follows directly from the fact that the specific cases of t-approval-Bribery and Borda-Bribery are NP-hard.

Little work had been conducted on approximating Bribery. Of interest is Faliszewski's (2008) FPTAS for Plurality-Weighted-\$Bribery. In addition, Elkind et al. (2009) studied different scoring rules in the context of Swap- and Shift-Bribery. They provided a 2-approximation for Borda-Shift-Bribery, and in Elkind and Faliszewski's (2010) work it

was generalized to hold for all scoring rules. Finally, Faliszewski, Manurangsi, and Sornat (2019) provided a polynomial-time approximation scheme (PTAS) for all scoring rules.

6. Conclusions

At the center of this work there are two conceptual results. First, a new connection between Bribery and WCM was found. Specifically, we showed that approximating the *score margin* for SR-WCM translates to approximations on the overall *cost* for SR-Weighted-$Bribery. While a connection between the two problems is hardly surprising, we argue that the use of the min-score-margin objective for SR-WCM provides the missing ingredient for approximating SR-Bribery and its variants.

Second, we introduced the application of the BvN decomposition and related theorem to SR-WCM. Here the major combinatorial insight is that the theorem implies that when choosing ballots for each of the manipulators, it is sufficient to consider at most m^2 ballots—a polynomial number—of the $m!$ possible ballots, while still obtaining a fairly good approximation to the score margin.

Another takeaway is related to the power of LPs as a tool for bringing several constraints into consideration simultaneously. Indeed, SR-Bribery and its generalizations can be seen as a two stage process: voter elimination, followed by the addition of voters with new ballots. The former can be seen as a set cover variant, while the latter is exactly a coalitional manipulation instance. However, these problems should not be solved independently, and deciding which voters to bribe must be tightly integrated with the decision on their new strategy. We showed that this can be achieved by an LP: its fractional solution, determines both stages at once. While we cannot retain this property when requiring an integral solution, the LP still enables us (a) to take all information into account when deciding who to bribe, and (b) to induce an SR-WCM instance which does not add much to the overall cost.

We mention several open problems as further possible research directions:

- Some other voting rules are not positional scoring rules, but do have some intrinsic notion of a score; for instance, the Simpson and Copeland rules. Can methods similar to ours be used to provide approximations to $\mathcal{R}$-WCM and $\mathcal{R}$-Bribery variants where $\mathcal{R}$ is such a rule?

- Our bribery results excluded scoring rules which are neither non-concentrated nor constant. It would be interesting to better understand them. Can they be approximated? Alternatively, can we prove some hardness-of-approximation results with respect to such rules?

- Focusing on SR-UCM, can we find other approximation factor trade-offs w.r.t. α_1, m, and k? We note that a different approximation factor trade-off is provided in another work by the authors (Keller et al., 2019).

- For the case of SR-UCM, our algorithm provides an additive $\widetilde{O}(\alpha_1\sqrt{k})$-approximation to the score margin. On the other hand, for Borda-UCM, it can be rather easily proven that Zuckerman et al.'s (2009) greedy heuristic provides an additive $O(\alpha_1) = O(m)$-approximation to the score margin. While their method can be applied to scoring rules

other than Borda, it is not clear whether it provides any approximation guarantee in these cases. An immediate direction would be to prove such guarantees for these cases as well.

Acknowledgments

This work extends a previous conference paper (Keller et al., 2018). Research was done while the first author was a PhD student, under the supervision of the second author. This work was supported by the Israel Science Foundation, under Grant No. 1488/14 and Grant No. 1394/16. We are deeply indebted to the anonymous reviewers for their meticulous review and numerous helpful comments, which substantially improved the presentation.

Appendix A. Randomization, Probability, and Chernoff Bounds

In the paper we use various forms of concentration inequalities, which are inequalities that bound the probability of a random variable X deviating too far away from its expected value $\mathbb{E}[X]$, as a function of their distance $|X - \mathbb{E}[X]|$.

Chernoff Variant in Mitzenmacher and Upfal's (2005) Book. Let $X_1, \ldots, X_n$ be independent random variables in $[0, 1]$. Let $X = \sum_{i=1}^{n} X_i$. Then for $0 < \beta \leq 1$

$$\Pr[|X - \mathbb{E}[X]| \geq \beta \mathbb{E}[X]] \leq 2 \exp\left(-\frac{\beta^2 \mathbb{E}[X]}{3}\right) . \tag{19}$$

Also for general $\beta > 0$:

$$\Pr[X \geq (1 + \beta)\mathbb{E}[X]] \leq \exp\left(-\frac{\beta^2 \mathbb{E}[X]}{2 + \beta}\right) . \tag{20}$$

Remark. Equation (19) appears in Chapter 4 of Mitzenmacher and Upfal's (2005) book for the $\{0, 1\}$ case, which is generalized to the $[0, 1]$ case in Exercise 4.19 therein. Equation (20) is folklore.

We use the following corollary of the above Chernoff bounds, focusing on their asymptotic behavior for arbitrarily-chosen polynomially-small error-probabilities.

Lemma 1. *Let $X_1, \ldots, X_n$ be independent random variables where $X_i \in [0, u]$ for all i, λ be some constant, and $\mathcal{N}$ be some large enough value. Let $X = \sum_{i=1}^{n} X_i$. Then*

$$\Pr[X \notin [\mathbb{E}[X] \pm R_1(\lambda, u, \mathbb{E}[X])]] \leq \mathcal{N}^{-\lambda} ,$$

where $R_1(\lambda, u, \mathbb{E}[X]) = 6\lambda \max\{\sqrt{u\mathbb{E}[X]}, u\} \ln \mathcal{N}$ is the deviation we allow w.r.t. the expected value $\mathbb{E}[X]$.

Proof. Scale all random variables down by a factor of u, yielding $Y_i = X_i/u$ for all i and let $Y = X/u$. Notice that $\mathbb{E}[Y] = \mathbb{E}[X]/u$ as well. We split to cases:

- If $6\lambda u \ln \mathcal{N} \leq \mathbb{E}[X]$, let $\beta \leq 1$ be a value to be determined later. Then by eq. (19) w.r.t. Y, we obtain that,

$$\Pr\left[|X - \mathbb{E}[X]| \geq \beta \mathbb{E}[X]\right] = \Pr\left[|Y - \mathbb{E}[Y]| \geq \beta \mathbb{E}[Y]\right] \tag{21}$$

$$\leq 2 \exp\left(-\frac{\beta^2 \mathbb{E}[Y]}{3}\right) \tag{22}$$

$$= 2 \exp\left(-\frac{\beta^2 \mathbb{E}[X]}{3u}\right) . \tag{23}$$

  Setting $\beta = \sqrt{6\lambda u \ln \mathcal{N}} / \sqrt{\mathbb{E}[X]} \leq 1$, we get that

$$\Pr[|X - \mathbb{E}[X]| \geq 6\sqrt{\lambda u \mathbb{E}[X] \ln \mathcal{N}}] \leq \frac{1}{\mathcal{N}^\lambda} . \tag{24}$$

- Else then $6\lambda u \ln \mathcal{N} > \mathbb{E}[X]$. Let $\beta > 1$ be a value to be determined later. Then by eq. (20):

$$\begin{aligned}
\Pr\left[X - \mathbb{E}[X] \geq \beta \mathbb{E}[X]\right] &= \Pr\left[Y - \mathbb{E}[Y] \geq \beta \mathbb{E}[Y]\right] \\
&\leq \exp\left(-\frac{\beta^2 \mathbb{E}[Y]}{2 + \beta}\right) \\
&\leq \exp\left(-\frac{\beta^2 \mathbb{E}[Y]}{3\beta}\right) \\
&= \exp\left(-\frac{\beta^2 \mathbb{E}[X]}{3\beta u}\right) \\
&= \exp\left(-\frac{\beta \mathbb{E}[X]}{3u}\right) .
\end{aligned}$$

  Setting $\beta = 6\lambda u \ln \mathcal{N} / \mathbb{E}[X] > 1$, we obtain that

$$\Pr[X - \mathbb{E}[X] \geq 6\lambda u \ln \mathcal{N}] \leq \frac{1}{\mathcal{N}^\lambda} . \tag{25}$$

  For the symmetric case, it also holds that

$$\Pr\left[X - \mathbb{E}[X] < -6\lambda u \ln \mathcal{N}\right] = 0 ,$$

  since $X \geq 0 > \mathbb{E}[X] - 6\lambda u \ln \mathcal{N}$. Summing up:

$$\Pr[|X - \mathbb{E}[X]| \geq 6\lambda u \ln \mathcal{N}] \leq \frac{1}{\mathcal{N}^\lambda} . \tag{26}$$

The theorem follows from combining the two cases. □

Appendix B. The Birkhoff-von-Neumann Theorem

We use the BvN decomposition in order to weed out invalid ballots in the randomized rounding stage of our min-margin-SR-WCM algorithm, while at the same time reducing the size of the valid ballot search space from exponential to polynomial.

The BvN decomposition can be applied to doubly-stochastic matrices, as defined in Section 2. Compare their definition to a *permutation matrix*, in which there is a single 1 value in each row and in each column, all other values being 0.

As mentioned, every permutation matrix is also a doubly-stochastic matrix, and the BvN theorem supplies the link in the other direction. Roughly speaking, the Birkhoff-von Neumann theorem states that each doubly-stochastic matrix is a point in the Birkhoff polytope, whose vertices are all the $m!$ permutation matrices. In other words, every doubly-stochastic matrix can be obtained by a convex combination of all permutation matrices. Furthermore, a convex combination in which the number of nonzero coefficients is at most m^2 can be efficiently found. We repeat the theorem and supply its proof:

Theorem 2 (BvN Theorem)**.** *Let $\mathbf{Y} \in [0,1]^{m\times m}$ be a doubly-stochastic matrix. We can decompose $\mathbf{Y}$ to a convex combination of at most m^2 permutation matrices, that is, we decompose $\mathbf{Y} = \lambda_1\mathbf{P}^{\tau_1} + \cdots + \lambda_q\mathbf{P}^{\tau_q}$ where each τ_t is a permutation with $\mathbf{P}^{\tau_t}$ being its corresponding permutation matrix, each $\lambda_t \in [0,1]$ and $\sum_{t=1}^{q}\lambda_t = 1$, and $q \le m^2$. This decomposition can be found in polynomial time.*

Proof. Given $\mathbf{Y} = [y_{i,j}]$, repeat the following steps for every $t = 1, 2, \ldots$ until the first step fails:

1. Find a permutation τ_t such that $y_{\tau_t(j),j} > 0$ for all j (implementation will be described soon).

2. Let $\lambda_t = \min_j y_{\tau_t(j),j}$. Perform the update: $\mathbf{Y} \leftarrow \mathbf{Y} - \lambda_t\mathbf{P}^{\tau_t}$, where $\mathbf{P}^{\tau_t}$ is the permutation matrix corresponding to τ_t.

We then use the following lemma.

Lemma 25. *The above algorithm performs at most m^2 iterations.*

Proof. By noticing that in each step t, at least one entry of $\mathbf{Y}$ becomes zero, that is the entry $(\tau_t(j'), j')$ such that $j' = \arg\min_j y_{\tau_t(j),j}$. When all entries will be zeros, Step 1 will inevitably fail. □

Lemma 26. *Until matrix $\mathbf{Y}$ becomes the all-zeros matrix, there always exists a permutation τ such that $y_{\tau(j),j} > 0$ for all j. Furthermore, τ can be found by a polynomial-time algorithm.*

Proof. First notice that as we perform step 2, each row or column in the matrix $\mathbf{Y}$ is affected equally (loses $\lambda_t \cdot 1 = \lambda_t$ from its sum) and therefore $\mathbf{Y}$ always remains 'almost' doubly-stochastic: that is, the sums of each row and each column are equal, though they are no longer 1. For such a matrix, a permutation abiding the aforementioned requirements can always be found by using Hall's (1935) marriage theorem, as we will immediately show.

For $\mathbf{Y}$ at some point in time t, let s_t be the sum of each row (or column). Now define a bipartite graph $G = (U \cup V, E)$, where $U = \{u_1, \ldots, u_m\}$ is a vertex set representing the rows of $\mathbf{Y}$, $V = \{v_1, \ldots, v_m\}$ is a vertex set representing the columns of $\mathbf{Y}$, and there is an edge $(u_i, v_j) \in E$ if and only if $y_{i,j} > 0$. According to Hall's marriage theorem, there is a perfect matching in G if and only if for every subset $U' \subseteq U$, it holds that $|N(U')| \ge |U'|$, where $N(U')$ is the set of neighbors of the vertices in U'. To see that this condition indeed

holds, consider any subset $U' \subseteq U$ and notice that the sum $\sum_{i \in U'} \sum_{j=0}^{m-1} y_{i,j}$ of all entries of the rows in U' equals $s_t|U'|$. This sum can be written equivalently as $\sum_{i \in U'} \sum_{j \in N(U')} y_{i,j} = s_t|U'|$. However, since the sum of each column is s_t, each column can contribute at most s_t to this sum and thus in order to reach the $s_t|U'|$ value, the entries in the sum have to originate in at least $|U'|$ columns, therefore $|N(U')| \geq |U'|$. Now, by using Hall's marriage theorem, we get that there exists a perfect matching M in G. Since a perfect matching in a bipartite graph naturally defines a permutation $\tau \colon [m] \to [m]$ which maps a column index j to a row index i if $(u_i, v_j) \in M$, we have just found a permutation τ where $\tau(j) = i$ implies that $y_{i,j} > 0$. In addition, as a perfect matching in a bipartite graph (if one exists) can be found in polynomial time (Hopcroft & Karp, 1973), the lemma follows. □

Lemma 27. *It holds that $\lambda_t \in [0, 1]$ for all t and that $\sum_t \lambda_t = 1$.*

Proof. $\lambda_t \in [0, 1]$ since every chosen λ_t is always a positive value bounded by an entry in $\mathbf{Y}$, and the entries of $\mathbf{Y}$ are always less than or equal to 1. Fix an arbitrary row i in the original matrix $\mathbf{Y}$, and notice that:

$$\begin{aligned}\sum_t \lambda_t &= \sum_j \sum_{\substack{t \\ \tau_t(j)=i}} \lambda_t \\ &= \sum_j y_{i,j} \\ &= 1 \ .\end{aligned}$$

□

We conclude that the decomposition we described follows the requirements of the lemma, and that finding it involves at most m^2 invocations of a bipartite maximum matching algorithm, and therefore can be done in polynomial time. □

References

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

Betzler, N., Niedermeier, R., & Woeginger, G. J. (2011). Unweighted coalitional manipulation under the Borda rule is NP-hard. In *the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*, pp. 55–60. IJCAI/AAAI.

Birkhoff, G. (1946). Tres observaciones sobre el algebra lineal. *Universidad Nacional de Tucumán Rev. Ser. A*, *5*, 147–151.

Brelsford, E., Faliszewski, P., Hemaspaandra, E., Schnoor, H., & Schnoor, I. (2008). Approximability of manipulating elections. In *the 23rd AAAI Conference on Artificial Intelligence (AAAI 2008)*, pp. 44–49. AAAI Press.

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

Conitzer, V., & Walsh, T. (2016). Barriers to manipulation in voting. In Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.), *Handbook of Computational Social Choice*, pp. 127–145. Cambridge University Press.

Davies, J., Katsirelos, G., Narodytska, N., Walsh, T., & Xia, L. (2014). Complexity of and algorithms for the manipulation of Borda, Nanson's and Baldwin's voting rules. *Artificial Intelligence*, *217*, 20–42.

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

Elkind, E., Faliszewski, P., & Slinko, A. M. (2009). Swap bribery. In *the 2nd International Symposium on Algorithmic Game Theory (SAGT 2009)*, Vol. 5814 of *Lecture Notes in Computer Science*, pp. 299–310. Springer.

Faliszewski, P. (2008). Nonuniform bribery. In *the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pp. 1569–1572. IFAAMAS.

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

Faliszewski, P., Manurangsi, P., & Sornat, K. (2019). Approximation and hardness of shift-bribery. In *the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, pp. 1901–1908. AAAI Press.

Faliszewski, P., & Procaccia, A. D. (2010). AI's war on manipulation: Are we winning?. *AI Magazine*, *31*(4), 53–64.

Faliszewski, P., & Rothe, J. (2016). Control and bribery in voting. In Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.), *Handbook of Computational Social Choice*, pp. 146–168. Cambridge University Press.

Faliszewski, P., Skowron, P., & Talmon, N. (2017). Bribery as a measure of candidate success: Complexity results for approval-based multiwinner rules. In *the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*, pp. 6–14. ACM.

Gibbard, A. (1973). Manipulation of voting schemes: a general result. *Econometrica*, *41*(4), 587–601.

Hall, P. (1935). On representatives of subsets. *Journal of the London Mathematical Society*, *10*(1), 26–30.

Hemaspaandra, E., & Hemaspaandra, L. A. (2007). Dichotomy for voting systems. *Journal of Computer and System Sciences*, *73*(1), 73–83.

Hemaspaandra, E., & Schnoor, H. (2016). Dichotomy for pure scoring rules under manipulative electoral actions. In *the 22nd European Conference on Artificial Intelligence (ECAI 2016)*, Vol. 285 of *Frontiers in Artificial Intelligence and Applications*, pp. 1071–1079. IOS Press.

Hopcroft, J. E., & Karp, R. M. (1973). An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, *2*(4), 225–231.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, *4*(4), 373–396.

Keller, O., Hassidim, A., & Hazon, N. (2018). Approximating bribery in scoring rules. In *the 32nd AAAI Conference on Artificial Intelligence (AAAI 18)*, pp. 1121–1129. AAAI Press.

Keller, O., Hassidim, A., & Hazon, N. (2019). New approximations for coalitional manipulation in scoring rules. *Journal of Artificial Intelligence Research*, *64*, 109–145.

Kőnig, D. (2001). *Theorie der endlichen und unendlichen Graphen*. American Mathematical Society.

Lin, A. P. (2012). *Solving hard problems in election systems*. Ph.D. thesis, Rochester Institute of Technology.

Mitzenmacher, M., & Upfal, E. (2005). *Probability and computing - randomized algorithms and probabilistic analysis*. Cambridge University Press.

Peters, J. W., & Alcindor, Y. (2016). Hillary Clinton struggles to win back young voters from third parties. https://www.nytimes.com/2016/09/29/us/politics/hillary-clinton-millennials-third-party.html. Accessed: 2017-08-31.

Procaccia, A. D., & Rosenschein, J. S. (2007). Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research*, *28*, 157–181.

Put, T., & Faliszewski, P. (2016). The complexity of voter control and shift bribery under parliament choosing rules. *Transactions on Computational Collective Intelligence*, *23*, 29–50.

Satterthwaite, M. A. (1975). 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.

von Neumann, J. (1953). A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games*, *2*, 5–12.

Xia, L., Conitzer, V., & Procaccia, A. D. (2010). A scheduling approach to coalitional manipulation. In *the 11th ACM Conference on Electronic Commerce (EC 2010)*, pp. 275–284. ACM.

Zuckerman, M., Procaccia, A. D., & Rosenschein, J. S. (2009). Algorithms for the coalitional manipulation problem. *Artificial Intelligence*, *173*(2), 392–412.