Strategic Campaign Management in Apportionment Elections

Robert Bredereck[1] , **Piotr Faliszewski**[2] , **Michał Furdyna**[2] ,
Andrzej Kaczmarczyk[1] and **Martin Lackner**[3]
[1]Technische Universität Berlin, Chair of Algorithmics and Computational Complexity, Berlin, Germany
[2]AGH University, Kraków, Poland
[3]TU Wien, Institute of Logic and Computation, Vienna, Austria
{robert.bredereck, a.kaczmarczyk}@tu-berlin.de, faliszew@agh.edu.pl, furdyna.michal@gmail.com
lackner@dbai.tuwien.ac.at

Abstract

In parliamentary elections, parties compete for a limited, typically fixed number of seats. We study the complexity of the following *bribery*-style problem: Given the distribution of votes among the parties, what is the smallest number of voters that need to be convinced to vote for our party, so that it gets a desired number of seats. We also run extensive experiments on real-world election data and measure the effectiveness of our method.

1 Introduction

Apportionment (or, party-list) elections [Balinski and Young, 1982; Pukelsheim, 2014] are among the most common forms of electing parliaments, used, e.g., in Austria, New Zealand, Poland, Spain, Turkey, and a number of other countries. In such elections each voter names the party that he or she supports, the vote counts for each party are tallied, and—based on this data—an *apportionment method* is used to assign a number of seats to each party (there are also separate procedures that specify which particular party members enter the parliament, but we disregard this issue). Some countries, such as Poland, are partitioned into districts and hold separate apportionment elections in each; the results of these elections are then summed up. Other countries, such as Austria, have a single, nation-wide district.[1] Further, many countries use various specific rules, such as assigning seats only to those parties that obtain a given minimal fraction of votes (referred to as the threshold); we abstract away from these details.

1.1 Our Contribution

We ask for efficient algorithms for the following problem: Given parties' vote counts (either for a single district or for a family of districts) and a preferred party, what is the smallest number of votes that need to be moved from the other parties to the preferred one, so that it obtains a desired number of seats (e.g., one more seat than it originally had, or 50%+1 seats in the parliament). Historically, such problems—where we modify voters' preferences—fall into the category of *bribery* problems [Faliszewski *et al.*, 2009; Faliszewski and Rothe, 2015], but they also have other, more benign and more useful, interpretations. We mention the following two (already taken, e.g., by Elkind and Faliszewski [2010], Faliszewski et al. [2017], Xia [2012], and Bredereck et al. [2017]):

1. Consider a political campaign preceding a given election. Based on poll data, leaders of each party may wish to know which voters they should try to convince to vote for their party. By solving our problem, they find the most effective way of obtaining additional seats.
2. After the election, one may wish to check how close the parties were to obtaining additional seats. If this number is small, then it is a reason to request a post-election audit.

In addition to the problem of obtaining a given number of seats for the preferred party, we also consider the problem where we ask for the lowest cost of ensuring that this party has more seats than any other one (i.e., that this party is a *winner* of the election). We focus on the family of divisor apportionment methods (which includes the well-known D'Hondt and Sainte Laguë methods), and on the largest remainder method. As a border case, we also study the first-past-the-post (FPTP) method.[2]

We find that for the single-district setting, all our problems can be solved in polynomial time, but while for FPTP it suffices to run a simple greedy algorithm, the other methods require more involved approaches, based on dynamic-programming. For the multi-district case the complexity spectrum is more interesting: While the single-district algorithms can be used to obtain a given number of seats for the preferred party, the problem of getting the largest number of seats is NP-hard for all methods (except FPTP).

We complement our study by performing a series of experiments on election data from Austria and Poland. First, we ask how useful it is to use our optimal algorithms, as opposed to either using simple "bribing" strategies (such as stealing voters from the strongest/weakest opponent, or equally from all the parties) or simply motivating abstainers to vote for our party. We find that moving voters optimally can, indeed, be

[1]Although Austria has multiple electoral districts, for the final seat apportionment the country is treated as a single district.

[2]Under this method, the party that has the most votes in a given district gets all the seats. This method is not used for parliamentary elections, but, in essence, it captures the presidential elections in the United States, where each state holds its own election and the winner receives all the electoral votes (seats, in our language).

quite beneficial: Often the simple strategies need to convince 1.5-3 times more voters than the optimal one (and in some cases up to 12 times as many).

In our second experiment, we ask how the number of districts affects the easiness of changing the election result. As one may expect, we find that the more districts there are, the fewer votes one has to move to obtain additional seats for a given party. Yet, it is surprising that the percentage of voters that need to be moved so that a given party has a majority in a parliament is often much lower than one might expect.

1.2 Related Work

Apportionment methods are studied deeply in mathematics and political science (see, e.g., the classic texts of Balinski and Young [1982] and Pukelsheim [2014]), but much less so in computer science and within computational social choice (a branch of artificial intelligence that studies computational properties of elections). Nonetheless, apportionment can be viewed as a special case of approval-based multiwinner elections [Brill *et al.*, 2018; Lackner and Skowron, 2018b]. As a consequence, for single-district settings with few parties only, one could use FPT bribery algorithms of Faliszewski *et al.* [2017]. Yet, our algorithms are faster and more direct (for other means of manipulating approval-based multiwinner rules, see, e.g., [Yang, 2019; Lackner and Skowron, 2018a; Peters, 2018]).

The paper of Güney [2018] is most closely related to ours: The author studies a campaign management problem similar to ours (specifically for the D'Hondt rule), but instead of providing polynomial-time algorithms, he models it as a mixed integer linear program. The paper shows that Turkish parliamentary elections (which are based on a multi-district system) are susceptible to strategic campaign management.

Finally, Ostapenko et al. [2012] developed a loosely related game-theoretic model of campaign management.

2 Preliminaries

For an integer t, we write $[t]$ to denote the set $\{1, \dots, t\}$. Further, by $[t]_0$ we mean the set $[t] \cup \{0\}$. We use the Iverson bracket notation, i.e., for a logical expression P we write $[P]$ to mean 1 if P is true, and to mean 0 if P is false.

2.1 Apportionment

An *apportionment problem* for m parties, $P_1, \dots, P_m$, is given by a number k of seats to allocate and a vote distribution $\mathbf{p} = (p_1, \dots, p_m)$ with $\sum_{i=1}^{m} p_i = n$ (where n is the number of voters; for each i, party P_i gets p_i votes). An *apportionment method* assigns seats to parties, i.e., it outputs a length-m sequence of nonnegative integers $\mathbf{a} = (a_1, \dots, a_m)$ with $\sum_{i=1}^{m} a_i = k$; such a sequence is called a *k-apportionment*. In case of ties it may output more than one apportionment (in our computational problems we will assume a particular tie-breaking).

In this paper we mostly consider *divisor methods* (with a focus on the D'Hondt and Sainte Laguë methods), the Largest Remainder (LRM) method, and the First-Past-The-Post (FPTP) method. Each divisor method is defined by an (infinite), increasing *divisor sequence* $d(1), d(2), d(3), \dots$.

D'Hondt				**Sainte Laguë**			
$d(\cdot)$	P_1	P_2	P_3	$d(.)$	P_1	P_2	P_3
1	3.0	**6.0**	**11.0**	1	**3.0**	**6.0**	**11.0**
2	1.5	3.0	**5.5**	3	1.0	2.0	**3.7**
3	1.0	2.0	**3.7**	5	0.6	1.2	2.2
4	0.8	1.5	2.8	7	0.4	0.9	1.6

Table 1: Division sequences for Example 1.

We require that each value $d(j)$ is computable in polynomial time with respect to j and is polynomially bounded in j (indeed, the standard divisor methods use very simple divisor sequences). Given an apportionment problem $\mathbf{p} = (p_1, \dots, p_m)$ with k seats to assign, to compute the apportionment, we calculate for each party P_i its *division sequence* $(p_i/d(1), p_i/d(2), \dots, p_i/d(k))$. Then, we find the largest number within these m division sequences and assign one seat to the corresponding party. Next we find the second largest number, assign one seat to its corresponding party, and continue in this fashion until all seats are assigned. In case of a tie, there exists more than one correct seat assignment.

Divisor methods can alternatively (and more compactly) be described as follows: Let x be the k-th largest entry *from the division sequences*. The divisor method defined by $d(1), d(2), \dots$ returns all k-apportionments $\mathbf{a} = (a_1, \dots, a_m)$ which satisfy ($p_i/d(0)$ is interpreted as ∞):

$$\frac{p_i}{d(a_i)} \geq x \quad \text{and} \quad \frac{p_i}{d(a_i+1)} \leq x \quad \text{for all } i \leq m.$$

The *D'Hondt method* (also known as the Jefferson method) is defined by the divisor sequence $1, 2, 3, \dots$; the *Sainte Laguë method* (also known as the Webster method) is defined by the divisor sequence $1, 3, 5, \dots$. There are also further divisor methods, such as Huntington–Hill, Adams, and Dean [Balinski and Young, 1982], but we do not focus on them.

The *Largest Remainder (LRM) method* (also known as the Hamilton method) assigns each party P_i at least $\mathrm{lqu}(p_i, n, k) = \lfloor \frac{p_i}{n} \cdot k \rfloor$ seats (we refer to them as *lower quota seats*); the remaining seats are distributed to the parties with the largest remainder values $\mathrm{rem}(p_i, n, k) = \frac{p_i}{n} \cdot k - \lfloor \frac{p_i}{n} \cdot k \rfloor$ (if there are r seats left to be assigned, then r parties with the highest remainder values get one each; we refer to these seats as *remainder seats*).

Finally, *First-Past-The-Post (FPTP)* gives all seats to the strongest party. This method is not an apportionment method (as it is not proportional), but we consider it as a corner case and due to its use in US presidential elections.

Example 1. *Let us consider an apportionment problem with three parties P_1, P_2, and P_3 and a vote distribution $\mathbf{p} = (3, 6, 11)$. Assume we want to fill 4 seats. For D'Hondt and Sainte Laguë, we select the 4 largest values in the corresponding divisor sequences (see Table 1), and obtain for D'Hondt the seat distribution $\mathbf{a} = (0, 1, 3)$, for Sainte Laguë $\mathbf{a} = (1, 1, 2)$. For LRM, P_1 receives $\mathrm{lqu}(3, 20, 4) = \lfloor 0.6 \rfloor = 0$ lower quota seats, P_2 receives $\mathrm{lqu}(6, 20, 4) = \lfloor 1.2 \rfloor = 1$, and P_3 receives $\mathrm{lqu}(11, 20, 4) = \lfloor 2.2 \rfloor = 2$ lower quota seats. The one remainder seat is assigned to P_1 as it has the largest remainder (0.6), and so LRM yields $\mathbf{a} = (1, 1, 2)$.*

2.2 Apportionment Bribery

In the following, we assume that apportionment methods are resolute, i.e., they employ a tie-breaking scheme and return a single apportionment. Specifically, we consider lexicographic tie-breaking, where ties are broken in favor of the party with the lowest index (this simplifies the formulation of some of our algorithms; our results can easily be adapted to other tie-breaking orders). Without loss of generality, we assume that we want to improve the result of party P_1. Given a vote distribution $\mathbf{p} = (p_1, \dots, p_m)$, we say that a vote distribution $\mathbf{q} = (q_1, \dots, q_m)$ is a $\mathbf{p}$-*valid bribery* if (i) $q_i \le p_i$ for all $i \in \{2, \dots, m\}$ (only party 1 gains votes) and (ii) $\sum_{i=1}^{m} q_i = n$ (no new voters appear). The *cost* of a valid bribery is defined as $c(\mathbf{p}, \mathbf{q}) = \sum_{i=2}^{m} p_i - q_i$; our model could easily be generalized to allow more complex cost functions.

Let $\mathcal{R}$ be an apportionment method and ℓ a positive integer.

$\mathcal{R}$ BRIBERY	
Instance:	An apportionment problem $\mathbf{p} = (p_1, \dots, p_m)$ for parties $P_1, \dots, P_m$, the number k of seats to allocate, a desired number ℓ of seats for party P_1, and a positive integer B (the bribing budget).
Question:	Is there a $\mathbf{p}$-valid bribery $\mathbf{q}$ such that $c(\mathbf{p}, \mathbf{q}) \le B$ and party 1 receives at least ℓ seats according to apportionment method $\mathcal{R}$?

This problem can be slightly modified, to ask for a vote distribution (within the specified bribing budget) so that party 1 receives more seats than each of the other parties ($a_1 > a_i$ for $i \ge 2$); we call this problem $\mathcal{R}$ WINNER BRIBERY.

Example 2. *Let us consider the* D'HONDT BRIBERY *problem with the apportionment instance from Example 1. Party 1, with 3 votes, can gain one more seat by convincing a single voter from P_2 to vote for P_1; D'Hondt yields apportionment* $\mathbf{a} = (1, 1, 2)$ *for the vote distribution* $(3 + 1, 6 - 1, 11)$. *Further, party 1 can gain two seats by moving 4 voters of P_3 to P_1, because D'Hondt yields apportionment* $\mathbf{a} = (2, 1, 1)$ *for the vote distribution* $(3 + 4, 6, 11 - 4)$ *(this result uses tie-breaking in favor of P_1). As P_1 ends up with more seats than the other parties, it means that for $B = 4$ we have a yes-instance of* D'HONDT WINNER BRIBERY.

We generalize $\mathcal{R}$ BRIBERY to multiple districts:

MULTI-DISTRICT $\mathcal{R}$ BRIBERY	
Instance:	A number of districts δ, δ apportionment problems for m parties, where for each $j \in [\delta]$, the j-th apportionment problems is given by $\mathbf{p}^j = (p_1^j, \dots, p_m^j)$ and k_j (the number of seats for this district); a desired number ℓ of seats for party 1, a budget $B \ge 0$.
Question:	Is there, for every $1 \le j \le \delta$, a $\mathbf{p}^j$-valid bribery $\mathbf{q}^j$ such that $\sum_{j=1}^{\delta} c(\mathbf{p}^j, \mathbf{q}^j) \le B$ and party 1 receives in total at least ℓ seats according to apportionment method $\mathcal{R}$ applied to each district individually?

As before, we also consider the MULTI-DISTRICT $\mathcal{R}$ WINNER BRIBERY problem, where we ask for P_1 to have more seats than each of the other parties.

We adopt the following important convention regarding our problems: We assume that the numbers of voters (and, thus, the bribing budgets) are specified in unary. We do so, because each vote corresponds to a single person who cast it and thus this number is of reasonable size (typically $< 10^9$). Further, each vote is stored physically as a separate ballot. Nonetheless, dropping this assumption might be interesting for future work as it may increase the complexity of our problems.

3 Algorithms for the Single-District Case

We start the discussion by considering the single-district cases, i.e., by considering the $\mathcal{R}$ BRIBERY and $\mathcal{R}$ WINNER BRIBERY problems. We find polynomial time algorithms for all our apportionment methods, for both these problems. As a warm-up, let us consider the FPTP method, for which a simple greedy algorithm suffices for both the problems.

Proposition 1. FPTP BRIBERY *and* FPTP WINNER BRIBERY *are both in* P.

Unfortunately, greedy algorithms do not seem to work for the other apportionment methods and, indeed, we resort to dynamic programming in their cases. As a consolation prize, we find algorithms that are nearly identical for the cases of divisor methods and LRM.

Theorem 1. *Let $\mathcal{R}$ be a divisor method with divisor sequence $d(1), d(2), \dots$ or the LRM method. The $\mathcal{R}$* BRIBERY *problem is in* P.

Proof. Let $(p_1, \dots, p_m)$ be our input vote distribution, let k be the number of seats to be allocated, of which we want P_1 to get at least ℓ, and let B be the bribing budget (we assume that $B < p_2 + \dots + p_m$ as otherwise we would move all the votes to P_1). We first take $\mathcal{R}$ to be a divisor method.

By the definition of our problem (and by monotonicity of divisor methods) we move exactly B votes to party P_1, so, in the end, party P_1 will have $p_1 + B$ votes. The main part of our algorithm is to compute from which parties we need to take these B votes so that P_1 ends up with at least ℓ seats. To this end, for each nonnegative integer p we define $\mathrm{seats}(p)$ to be the nonnegative integer λ such that:

$$\left(\frac{p}{d(\lambda)} > \frac{p_1 + B}{d(\ell)}\right) \wedge \left(\frac{p}{d(\lambda + 1)} \le \frac{p_1 + B}{d(\ell)}\right),$$

except that if ${p}/{d(1)} \le {p_1+B}/{d(\ell)}$ then $\mathrm{seats}(p) = 0$, and if $\frac{p}{d(k)} > \frac{p_1+B}{d(\ell)}$ then $\mathrm{seats}(p) = k$. Intuitively, $\mathrm{seats}(p)$ is the number of seats that a party (other than P_1) gets until P_1 gets its ℓ-th seat, provided that this party ends up with p votes.

Our goal is to decide if there is a sequence of nonnegative integers $(b_2, \dots, b_m)$, specifying the numbers of voters that we move from respective parties to P_1, such that $\sum_{j=2}^{m} b_j = B$ (so, indeed, we move B voters) and $\sum_{j=2}^{m} \mathrm{seats}(p_j - b_j) \le k - \ell$ (so, indeed, P_1 is assigned the ℓ-th seat, because there are enough seats left for this to happen). To check if such a sequence exists, we use a dynamic programming approach.

For each $i \in [m] \setminus \{1\}$, $s \in [k]_0$, and $b \in [B]_0$, let $f(i, s, b)$ be a Boolean function whose value is *true* exactly if there is a sequence $(b_2, \dots, b_i)$ of nonnegative integers such that

$\sum_{j=2}^{i} b_j = b$ and $\sum_{j=2}^{i} \mathrm{seats}(p_j - b_j) = s$. Our algorithm accepts exactly if $\bigvee_{k' \in [k-\ell]_0} f(m, k', B)$ is *true*.

To compute the values $f(i, s, b)$ in polynomial time, we first set $f(1, 0, 0)$ to be *true* and we set $f(1, s, b)$ to be *false* for all $(s, b) \neq (0, 0)$. Then, we note that for each $i \geq 2$ we have:

$$f(i, s, b) = \bigvee_{b_i \in [b]_0} f(i-1, s - seats(p_i - b_i), b - b_i).$$

Using this recursion and standard dynamic-programming techniques, we obtain a polynomial-time algorithm for computing values $f(i, j, b)$. This completes the proof for the case of divisor methods.

For the case of LRM we proceed as follows. First, we note that there is no reason to move fewer than B votes to P_1. Thus, in the final election P_1 will get $q_1 = \mathrm{lqu}(p_1 + B, n, k)$ lower quota seats and, possibly, one additional remainder seat. If $q_1 \geq \ell$ then we immediately accept. Similarly, if $q_1 < \ell - 1$ then we immediately reject. Otherwise, if P_1 would get exactly $\ell - 1$ lower quota seats, we need to check if there is a way of moving B votes from the other parties so that P_1 gets its remainder seat. To this end, we use the same function f as for the divisor methods, except that we redefine the meaning of $\mathrm{seats}(p)$ to be:

$$\mathrm{lqu}(p, n, k) + [\mathrm{rem}(p, n, k) > \mathrm{rem}(p_1 + B, n, k)].$$

The interpretation of $\mathrm{seats}(p)$ is as follows: Consider a party (other than P_1) that has p votes. Then, $\mathrm{seats}(p)$ gives the number of lower quota seats that this party gets plus the number of remainder seats (0 or 1) that it is guaranteed to get if P_1 gets a remainder seat (recall that tie-breaking is in favor of P_1). As in the divisor methods case, our algorithm accepts exactly if $\bigvee_{k' \in [k-\ell]_0} f(m, k', B)$ is *true*. □

Similar algorithms also work for the cases of ensuring that P_1 has strictly more seats than any other party (but without requiring a particular number of them), i.e., for WINNER BRIBERY. However, in this scenario the algorithms require more care. The issue is that while our algorithms for BRIBERY find ways to give ℓ seats to party P_1 (provided that this is possible), as a side effect they may also increase the numbers of seats allocated to other parties. It is also possible that to solve WINNER BRIBERY it suffices to move votes in such a way that a party with the most seats loses some of them in favor of parties other than P_1. Finally, we need to pay more attention to tie-breaking.

Theorem 2. *Let $\mathcal{R}$ either be a divisor method or LRM. The $\mathcal{R}$* WINNER BRIBERY *problem is in* P.

To conclude, we note that our algorithms from Theorems 1 and 2 can be adapted to the case where for each party P_i, $i \in \{2, \ldots, m\}$, there is a polynomially-bounded, nondecreasing function $\mathrm{cost}_i(x)$ (provided as part of the input), specifying the cost of moving x voters from P_i to P_1, and where we ask if there is a bribery of total cost at most B. This way we can, e.g., model the increasing difficulty of convincing larger groups of voters to vote for P_1. To take such functions into account, in our algorithms we would modify the f function to not return a *true/false* value, but the lowest cost of achieving a particular effect (where cost ∞ would correspond to impossibility).

4 Algorithms for the Multi-District Case

It turns out that we can use the single-district algorithms to solve the MULTI-DISTRICT $\mathcal{R}$ BRIBERY problem. Briefly put, we can compute the cost of getting each possible number of seats for P_1 in each district separately, and then solve a Knapsack-like problem to find out if by moving at most B voters we can obtain ℓ seats for P_1 (which we can do in polynomial time due to our assumption that B is given in unary).

Proposition 2. *Let $\mathcal{R}$ either be a divisor method with divisor sequence $d(1), d(2), \ldots$ or LRM. The* MULTI-DISTRICT $\mathcal{R}$ BRIBERY *problem is in* P.

On the other hand, MULTI-DISTRICT $\mathcal{R}$ WINNER BRIBERY is NP-hard. Intuitively, this is so because the problem gives us flexibility to require that all parties from a given set lose one seat each, and form districts so that in each of them only subsets of these parties can lose seats. This way we form a reduction from CUBIC VERTEX COVER.

Theorem 3. *Let $\mathcal{R}$ either be a divisor method with divisor sequence $d(1), d(2), \ldots$ or LRM.* MULTI-DISTRICT $\mathcal{R}$ WINNER BRIBERY *is* NP*-hard, even if the number of seats per districts as well as the number of voters transferred per district is at most three and the total number of votes per district is a constant depending on $d(1)$ and $d(2)$.*

Proof. We show NP-hardness by a reduction from the CUBIC VERTEX COVER problem. An instance of CUBIC VERTEX COVER consists of a graph G, where each vertex has exactly three neighbors, and an integer k; we ask if it is possible to choose k vertices so that each edge is incident to at least one of them. Below we provide a reduction for the case of the D'Hondt apportionment method (we omit the details regarding how to adapt it for other methods).
Construction. Let (G, k) be our input instance of CUBIC VERTEX COVER, where $G = (V, E)$ is a graph and k is an integer (without loss of generality, we assume that $k \geq 3$). Further, let $V = \{v_1, \ldots, v_r\}$ and $E = \{e_1, \ldots, e_t\}$. We form an instance of MULTI-DISTRICT D'HONDT WINNER BRIBERY as follows. First, we set our bribing budget to be $B = 3k$. Next, we create $1 + t + 10r + 1$ parties:

1. P_1 is our preferred party.
2. For each edge e_i, we form an *edge party* $P(e_i)$=P_{1+i}
3. For each vertex v_j we form 10 *dummy parties*, so for each $\ell \in [10]$ we let $P(v_j, \ell) = P_{1+t+10(j-1)+\ell}$ be the ℓ-th dummy party for vertex v_j.
4. We also form one *blocker* party $P(b) = P_{1+t+10r+1}$.

Then we form $r + t(k-2) + (k-1)$ districts:

1. For every vertex v_j we form a *vertex district* $d(v_j)$ with three seats to allocate, where P_1 gets 7 votes, each dummy party associated with v_j gets 10 votes, and for each edge e_i incident to v_j, party $P(e_i)$ gets 10 votes.

2. We form $t(k-2)$ dummy districts, each with a single seat to allocate. For each edge party there are exactly $k-2$ dummy districts where this party gets $2B+1$ votes and all the other parties get zero votes (this way this edge party gets the seat and a bribery of cost at most B cannot change that).
3. We form $k-1$ blocker districts, each with a single seat to allocate. In each of these districts the blocker party gets $2B+1$ votes and all the other parties get zero votes.

This completes the description of our construction.

Initial Seat Allocation. Let us now describe the initial seat allocation, prior to any bribery. The blocker party gets exactly $k-1$ seats from the blocker district (it has zero votes in every other district so it cannot get seats anywhere else). Similarly, each edge party gets $k-2$ seats from the dummy districts. For the vertex districts, let us recall that the tie-breaking prefers party P_1 over the edge parties, which are then preferred over the dummy parties and the blocker party. Now consider some vertex district $d(v_j)$, and let e_a, e_b, and e_c be the three edges incident to v_j. There are three seats to allocate and one can verify that, due to tie-breaking, each of the parties $P(e_a)$, $P(e_b)$, and $P(e_c)$ gets one of them. In total, party P_1 and the dummy parties get no seats, the blocker party gets $k-1$ seats, and the edge parties get k seats each. In particular, P_1 does not have the majority of seats in the unbribed election.

Main Idea. The intuition behind our construction is that with budget $B = 3k$, P_1 can obtain k seats by getting one seat in k vertex districts. We ensure that this is enough to have more seats than any other party exactly if the k districts correspond to a vertex cover in G.

Correctness. To show the correctness of our reduction, we will show that graph G has a vertex cover of size k if and only if there is a valid bribery that moves at most B voters and ensures that party P_1 gets more seats than any other party.

For the "only if" direction, assume that G has a vertex cover of size k. In particular, let S be a set of k vertices that forms a vertex cover for G. We form a bribery as follows. For each vertex district that corresponds to a vertex $v \in S$, we transfer to P_1 one vote from each of the three edge parties corresponding to edges incident to v. In every such district, P_1 now has 10 votes, every edge party has at most 9 votes, and the dummy parties still have 10 votes each. Thus, due to tie-breaking, in each such district one seat goes to P_1 and the other two seats go to the dummy parties (each of them to a different dummy party). It directly follows that this gives P_1 k seats. Moreover, since every edge is covered by at least one vertex from S, every edge party looses at least one seat and ends up with $k-1$ seats. The blocker party also gets $k-1$ seats, so P_1 has more seats than any other party.

We omit the "if" direction due to limited space. $\square$

It is also interesting to consider the parametrized complexity of MULTI-DISTRICT $\mathcal{R}$ WINNER BRIBERY. We show that the problem is W[1]-hard for the parameterization by the number of districts.

Theorem 4. *Let $\mathcal{R}$ be a divisor method with divisor sequence $d(1), d(2), \ldots$ or the LRM method. The* MULTI-DISTRICT $\mathcal{R}$ WINNER BRIBERY *is* W[1]*-hard with respect to the number of districts.*

We have not been able to find a hardness reduction nor to find an FPT algorithm for the parameter "number of parties" (a natural approach would be, e.g., to form an integer linear program and solve it using an FPT algorithm; unfortunately, for Lenstra's algorithm [Lenstra, Jr., 1983] we end up with too many variables, and for many other approaches, including n-fold IP, we end up with two large coefficients in the constraints; see the overview of Gavenciak et al. [2018]).

5 Experiments

With our experimental analysis, we approach two main questions: First, we ask how much more effective is an optimal bribing strategy as opposed to simpler heuristics. Second, we study the influence of the number of districts on the easiness of changing the election result. We use two datasets, vote counts from Polish parliamentary elections from years 2011, 2015, and 2019, with 41 districts each; and Austrian parliamentary elections from years 1994–2019 (nine single-district elections). These two countries were chosen as they both use the D'Hondt method.

For a given vote distribution $\mathbf{p}$, party P_i, and desired number of seats ℓ, we express the *effectiveness* of a bribing strategy S as follows. Let x be the smallest number such that after adding x votes supporting P_i, party P_i gets (at least) additional ℓ seats. Then, let y be the number of votes that S moves to P_i to obtain ℓ additional seats. We define the effectiveness of S as x/y. Intuitively, this indicates how much more effective it is to convince the voters who already decided to vote relative to bringing in new ones. We consider the following bribing strategies for party P_i:

1. *optimal bribery*: we move the voters from other parties to P_i optimally (as in the algorithms from Sections 3 and 4, but using P_i in the place of P_1).
2. *weakest/strongest rival*: move voters from the weakest/strongest party to P_i (and if this rival party does not have enough voters, then introduce additional voters),[3]
3. *balanced bribery*: move votes from the other parties proportionally to their vote counts (this is a derandomized analogue of bribing voters uniformly at random).

One can view the latter two strategies as simple heuristics for solving the bribing problem.

Table 2 shows the average efficiency for gaining one seat in the Polish and Austrian elections. We distinguish values for all parties on average (1st column), for the strongest party, and for the weakest party. One can see that the strongest party is generally in the best position to benefit from moving voters. Further, optimal bribery is significantly more effective than the other forms. When choosing between the simple bribing strategies, taking votes from the strongest rival appears to be the most promising option: a vote moved from the strongest rival is roughly 1.5 times as effective as an additional, previously abstaining, voter.

[3]By the strongest/weakest party we mean the one that originally had the most/the fewest votes.

		average	strongest party	weakest party
Poland	optimal bribery	2.32	2.77	2.38
	balanced bribery	1.39	1.53	1.43
	from weakest rival	1.27	1.37	1.21
	from strongest rival	1.50	1.62	1.61
Austria	optimal bribery	1.85	2.30	1.73
	balanced bribery	1.29	1.41	1.16
	from weakest rival	1.20	1.31	1.04
	from strongest rival	1.35	1.51	1.03

Table 2: Effectiveness of different bribing strategies for one additional seat (Austria and Poland, both use D'Hondt)

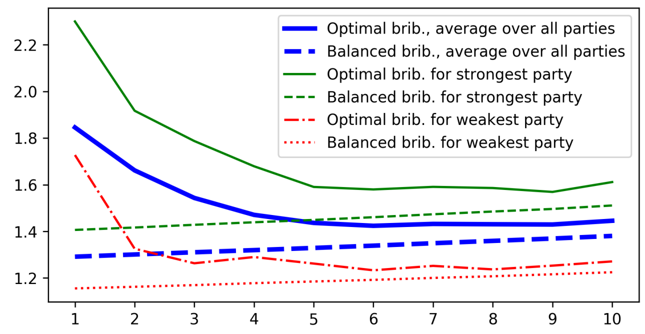

Figure 1: Average effectiveness of optimal and balanced bribery in the Austrian dataset; the x-axis shows the number of seats gained.

Let us now move on to Figure 1, which compares effectiveness values for an increasing number of seats to be gained (for the Austrian dataset; the Polish data yields qualitatively similar results, but generally with higher effectiveness of bribery). We see that as the number of additional seats increases, the effectiveness of optimal bribery diminishes, and becomes similar to that of balanced bribery (although there still is some advantage to using optimal bribery). This is due to the proportional nature of D'Hondt; larger changes require a roughly proportional amount of votes.

While the most effective optimal bribery in the Austrian dataset had an efficiency of 3.7, the most effective optimal bribery in the Polish dataset had one of 12.6. Together with the insights gained from Table 2, this encourages the hypothesis that fewer districts decrease the potential gain from moving voters. We investigate this now in more detail by presenting a different empirical study on the Polish election data.

Here, we consider the Polish elections and compute for each party an optimal bribery allowing this party to obtain a majority ($50\%+1$) of seats. To study the effect of the number of districts, we start with the original partition of the country into 41 electoral districts, and then we decrease their number by merging districts. We do this sequentially, always merging two districts chosen uniformly at random, until only one large district remains. The result for a given number of districts is computed as follows: (1) we take the average of the number of bribed voters over five trials, each for a different districting (computed as described above); (2) we compute the average number of vote transfers per gained seat, and (3) we divide it by the overall number of voters. This way we can compare the results for different initial elections, obtaining an average "relative price" per seat.

# districts	41	30	20	1
Moves per Seat/# voters	0.12%	0.14%	0.15%	0.21%

Table 3: The results for party 2 and the Polish 2019 election; the results are representative for all conducted experiments.

We conducted the experiment for all three elections in the Polish dataset where there are 460 seats in total, six parties, and between 12 and 18 million voters. For each election, we considered some parties 1, 2, and 3 from the ballots used in these elections (the numbers are assigned to parties uniformly at random, prior to each election); we excluded party 3 for the election from 2019 since it already initially got a majority of seats. Despite the fact that we considered different parties in different elections, and despite the fact that they received very different vote counts, the results we obtained were qualitatively the same. Thus, in Table 3, we provide the results for party 2 and the election from 2019 for the original 41 districts, 30 districts, 20 districts, and a single district.

The results of this experiment confirm that the fewer districts there are, the more vote moves are required to obtain a certain number of seats. To be precise, when we merged half of the districts, then, on average, we needed around 22% more vote moves per gained seat. This is natural as with fewer districts, there are fewer "rounding errors" that apportionment needs to deal with. It is, however, interesting that the results do not differ much, no matter if a party needed around 220 or around 100 additional seats (out of 460) to get a majority. What is particularly striking is that even for a party with few seats (11 in our case) at the beginning, it was enough to move only around 26% of the voters to obtain the majority.

6 Future Work

We study election campaign management (modeled as bribery) in the apportionment setting, focusing on the computational perspective. Our algorithms require precise information about vote counts and, as this information is difficult to obtain *before* an election, we ask how such campaigns can be planned given only approximate vote counts. The comparison of simple bribing schemes in Section 5 can be seen as the first step in this direction. Further, we focused on popular apportionment schemes, but there are others such as, e.g., the Quota method [Balinski and Young, 1975] or Frege's apportionment method [Harrenstein *et al.*, 2019].

Acknowledgements

We are grateful to anonymous reviewers for their insightful comments. AK was supported by the DFG project "AFFA" (BR 5207/1 and NI 369/15). PF was supported by the Friedrich Wilhelm Bessel Award from the Alexander von Humboldt Foundation. ML was supported by the Austrian ScienceFund (FWF): P31890. Many ideas for this work arose during Dagstuhl seminar 19381.

References

[Balinski and Young, 1975] M. Balinski and H. P. Young. The quota method of apportionment. *The American Mathematical Monthly*, 82(7):701–730, 1975.

[Balinski and Young, 1982] M. Balinski and H. P. Young. *Fair Representation: Meeting the Ideal of One Man, One Vote*. Yale University Press, 1982.

[Bredereck *et al.*, 2017] R. Bredereck, P. Faliszewski, A. Kaczmarczyk, R. Niedermeier, P. Skowron, and N. Talmon. Robustness among multiwinner voting rules. In *Proceedings of SAGT-17*, pages 80–92, 2017.

[Brill *et al.*, 2018] M. Brill, J. Laslier, and P. Skowron. Multiwinner approval rules as apportionment methods. *Journal of Theoretical Politics*, 30(3):358–382, 2018.

[Elkind and Faliszewski, 2010] E. Elkind and P. Faliszewski. Approximation algorithms for campaign management. In *Proceedings of WINE-10*, pages 473–482, 2010.

[Faliszewski and Rothe, 2015] P. Faliszewski and J. Rothe. Control and bribery in voting. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 7. Cambridge University Press, 2015.

[Faliszewski *et al.*, 2009] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. How hard is bribery in elections? *Journal of Artificial Intelligence Research*, 35(1):485–532, 2009.

[Faliszewski *et al.*, 2017] P. Faliszewski, P. Skowron, and N. Talmon. Bribery as a measure of candidate success: Complexity results for approval-based multiwinner rules. In *Proceedings of AAMAS-17*, pages 6–14, 2017.

[Gavenciak *et al.*, 2018] T. Gavenciak, D. Knop, and M. Koutecký. Integer programming in parameterized complexity: Three miniatures. In *Proceedings of IPEC-18*, pages 21:1–21:16, 2018.

[Güney, 2018] E. Güney. A mixed integer linear program for election campaign optimization under D'Hondt rule. In *Proceedings of OR-17*, pages 73–79, 2018.

[Harrenstein *et al.*, 2019] P. Harrenstein, M.-L. Lackner, and M. Lackner. A mathematical analysis of an election system proposed by Gottlob Frege. *arXiv preprint arXiv:1907.03643*, 2019.

[Lackner and Skowron, 2018a] M. Lackner and P. Skowron. Approval-based multi-winner rules and strategic voting. In *Proceedings of IJCAI-18*, pages 340–436, 2018.

[Lackner and Skowron, 2018b] M. Lackner and P. Skowron. Consistent approval-based multi-winner rules. In *Proceedings of EC-18*, pages 47–48, 2018.

[Lenstra, Jr., 1983] H. Lenstra, Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

[Ostapenko *et al.*, 2012] V. Ostapenko, O. Ostapenko, E. Belyaeva, and Y. Stupnitskaya. Mathematical models of the battle between parties for electorate or between companies for markets. *Cybernetics and Systems Analysis*, 48(6):814–822, 2012.

[Peters, 2018] D. Peters. Proportionality and strategyproofness in multiwinner elections. In *Proceedings of AAMAS-18*, pages 1549–1557, 2018.

[Pukelsheim, 2014] F. Pukelsheim. *Proportional Representation: Apportionment Methods and Their Applications*. Springer, 2014.

[Xia, 2012] L. Xia. Computing the margin of victory for various voting rules. In *Proceedings of EC-12*, pages 982–999, 2012.

[Yang, 2019] Y. Yang. Complexity of manipulating and controlling approval-based multiwinner voting. In *Proceedings of IJCAI-19*, pages 637–643, 2019.