Discrete Optimization 11 (2014) 1–21

Contents lists available at ScienceDirect

Discrete Optimization

journal homepage: www.elsevier.com/locate/disopt

The complexity of probabilistic lobbying☆,☆☆

Daniel Binkele-Raible [a], Gábor Erdélyi [b], Henning Fernau [a,*], Judy Goldsmith [c], Nicholas Mattei [d,e,1], Jörg Rothe [f]

[a] Universität Trier, Fachbereich 4—Abteilung Informatikwissenschaften, 54286 Trier, Germany
[b] University of Siegen, School of Economic Disciplines, 57068 Siegen, Germany
[c] University of Kentucky, Department of Computer Science, Lexington, KY 40506, USA
[d] NICTA, Sydney, Australia
[e] University of New South Wales, Sydney, Australia
[f] Heinrich-Heine-Universität Düsseldorf, Institut für Informatik, 40225 Düsseldorf, Germany

ARTICLE INFO

Article history:
Received 22 February 2012
Received in revised form 22 October 2013
Accepted 26 October 2013
Available online 22 November 2013

Keywords:
Computational complexity
Parameterized complexity
Approximability
Computational social choice

ABSTRACT

We propose models for lobbying in a probabilistic environment, in which an actor (called "The Lobby") seeks to influence voters' preferences of voting for or against multiple issues when the voters' preferences are represented in terms of probabilities. In particular, we provide two evaluation criteria and two bribery methods to formally describe these models, and we consider the resulting forms of lobbying with and without issue weighting. We provide a formal analysis for these problems of lobbying, and determine their classical and parameterized complexity depending on the given bribery/evaluation criteria and on various natural parameterizations. Specifically, we show that some of these problems can be solved in polynomial time, some are NP-complete but fixed-parameter tractable, and some are $W[2]$-complete. Finally, we provide approximability and inapproximability results for these problems and several variants.

1. Introduction

1.1. Motivation and informal description of probabilistic lobbying models

In most democratic political systems, laws are passed by elected officials who are supposed to represent their constituency. Individual entities such as citizens or corporations are not supposed to have undue influence in the wording or passage of a law. However, they are allowed to make contributions to representatives, and it is common to include an indication that the contribution carries an expectation that the representative will vote a certain way on a particular issue.

☆ A preliminary version of this paper appears in the proceedings of the *First International Conference on Algorithmic Decision Theory*, October 2009 (Erdélyi et al., 2009 [1]).

☆☆ This work was done in part while the second author was affiliated to Heinrich-Heine-Universität Düsseldorf and to Nanyang Technological University, Singapore, and visiting Universität Trier, while the first and the fourth author were visiting Heinrich-Heine-Universität Düsseldorf, the sixth author was visiting the University of Rochester and Stanford University, and the fifth author was affiliated to the University of Kentucky.

* Corresponding author. Tel.: +49 6512012827.
E-mail addresses: raible@informatik.uni-trier.de (D. Binkele-Raible), erdelyi@wiwi.uni-siegen.de (G. Erdélyi), fernau@uni-trier.de (H. Fernau), goldsmit@cs.uky.edu (J. Goldsmith), nicholas.mattei@nicta.com.au (N. Mattei), rothe@cs.uni-duesseldorf.de (J. Rothe).

1 NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

1572-5286/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.disopt.2013.10.003

Many factors can affect a representative's vote on a particular issue. There are the representative's personal beliefs about the issue, which presumably were part of the reason that the constituency elected them. There are also the campaign contributions, communications from constituents, communications from potential donors, and the representative's own expectations of further contributions and political support.

It is a complicated process to reason about. Earlier work (see the references given in Section 1.2) considered the problem of meting out contributions to representatives in order to pass a set of laws or influence a set of votes. However, the earlier computational complexity work by Christian et al. [2] on this problem made the assumption that a politician who accepts a contribution will in fact – if the contribution meets a given threshold – vote according to the wishes of the donor.

It is said that "An honest politician is one who stays bought", but that does not take into account the ongoing pressures from personal convictions and opposing lobbyists and donors. We consider the problem of influencing a set of votes under the assumption that we can influence only the *probability* that the politician votes as we desire.

There are several axes along which the picture is being complicated in realistic scenarios. To describe these formally, we introduce various evaluation criteria and bribery methods. The first one is the notion of sufficiency: What does it mean to say we have donated enough to influence the vote? Does it mean that the probability that a single vote will go our way is greater than some threshold? That the probability that all the votes go our way is greater than that threshold? We formally define and discuss these and other criteria in the section on evaluation criteria (Section 2.3).

The evaluation criteria we discuss are a proxy for what could be considered the most natural evaluation criteria: *probabilistic majority*. Ideally, a probabilistic win for an issue would be exactly when the sum of the probabilities of possible futures that include that win exceeds a given threshold. This would mean counting all possible future scenarios where a win was achieved. While we do not tackle this question directly in the current work, we study two useful relaxations of this winning criteria, which may be easier to express and to analyze algorithmically. In particular, we consider two methods for evaluating the outcome of a vote:

1. *strict majority*, where a vote on an issue is won by a strict majority of voters having a probability of accepting this issue that exceeds a given threshold, and
2. *average majority*, where a vote on an issue is won exactly when the voters' average probability of accepting this issue exceeds a given threshold.

How does one donate money to a campaign? In the United States there are several laws that influence how, when, and how much a particular person or organization can donate to a particular candidate. We examine ways in which money can be channeled into the political process in the section on bribery methods (Section 2.2). In particular, we consider two methods that an actor (called "The Lobby") can use to influence the voters' preferences of voting for or against multiple issues:

1. *microbribery*, where The Lobby may choose which voter to bribe on which issue in order to influence the outcome of the vote according to the evaluation criterion used and
2. *voter bribery*, where The Lobby may choose which voters to bribe and for each voter bribed the funds are equally distributed over all the issues, again aiming at changing the outcome of the vote according to the evaluation criterion used.

The voter bribery method is due to Christian et al. [2], who were the first to study lobbying in the context of direct democracy where voters vote on multiple referenda. Their "Optimal Lobbying" problem (denoted OL) is a deterministic and unweighted variant of the lobbying problems that we present in this paper. We state this problem in the standard format for parameterized complexity:

	Optimal Lobbying
Given:	An $m \times n$ 0/1 matrix E and a 0/1 vector $\vec{Z}$ of length n. Each row of E represents a voter and each column represents an issue. An entry of E is 1 if this voter votes "yes" for this issue, and is 0 otherwise. $\vec{Z}$ represents The Lobby's target outcome.
Parameter:	A positive integer b (representing the number of voters to be influenced).
Question:	Is there a choice of b rows of the matrix (i.e., of b voters) that can be changed such that in each column of the resulting matrix (i.e., for each issue) a strict majority vote yields the outcome targeted by The Lobby?

Christian et al. [2] proved that OL is $W[2]$-complete. Sandholm noted that the "Optimal Weighted Lobbying" (OWL) problem, which allows different voters to have different prices and so generalizes OL, can be expressed as and solved via the "binary multi-unit combinatorial reverse auction winner-determination problem" (see [3] for its definition).

The microbribery method in the context of lobbying – though inspired by the different notion of microbribery that Faliszewski et al. [4,5] introduced in the context of bribery in voting – is new to this paper.

Table 1
Complexity results for X-Y-PLP, where $X \in \{MB, VB\}$ and $Y \in \{SM, AM\}$.

Problem	Classical complexity	Parameterized complexity, parameterized by			Stated in or implied by Thm./Cor.
		Total budget	Budget per issue	Total budget & discretiz. level	
MB-SM-PLP	*P*	(FPT)	(FPT)	(FPT)	4.1
MB-AM-PLP	*P*	(FPT)	(FPT)	(FPT)	4.2
VB-SM-PLP	NP-complete	FPT	$W[2]$-complete	FPT	4.3 and 5.1–5.3
VB-AM-PLP	NP-complete	?	$W[2]$-hard	FPT	4.3, 5.4, 5.1

1.2. Related work

Lobbying has been studied formally by economists, computer scientists, and special interest groups since at least 1983 [6] and as an extension to formal game theory since 1944 [7]. The different disciplines have considered mostly disjoint aspects of the process while seeking to accomplish distinct goals with their respective formal models. Economists study lobbying as "economic games", as defined by von Neumann and Morgenstern [7]. This analysis is focused on learning how these complex systems work and deducing optimal strategies for winning the competitions [6,8,9]. This work has also focused on how to "rig" a vote and how to optimally dispense the funds among the various individuals [8]. Economists are interested in finding effective and efficient bribery schemes [8] as well as determining strategies for instances of two or more players [6,8,9]. Generally, they reduce the problem of finding an effective lobbying strategy to one of finding a winning strategy for the specific type of game. Economists have also formalized this problem for bribery systems in both the United States [6] and the European Union [10].

The study of lobbying from a computational perspective that was initiated by Christian et al. [2] falls into the field of computational social choice, which stimulates a bidirectional transfer between social choice theory (in particular, voting and preference aggregation) and computer science. For example, voting systems have been applied in various areas of artificial intelligence, most notably in the design of multiagent systems (see, e.g., [11]), for developing recommender systems [12], for designing a meta-search engine that aggregates the website rankings generated by several search engines [13], etc. Applications of voting systems in such automated settings (not restricted only to political elections in human societies) requires a better understanding of the computational properties of the problems related to voting. In particular, many papers have focused on the complexity of

- *winner determination* (see, e.g., [14–18]),
- *manipulation* (see, e.g., [5,19–32]),
- *procedural control* (see, e.g., [4,29,33–38]), and
- *bribery in elections* (see, e.g., [4,5]).

For more details, the reader is referred to the surveys by Faliszewski et al. [39–41], Conitzer [42], and Baumeister et al. [43] and the references cited therein. In comparison, much less work has been done on *lobbying in voting on multiple referenda* that we are concerned with here ([2], see also [3]).

1.3. Organization of this paper and a brief overview of results

Christian et al. [2] show that OL is complete for the (parameterized) complexity class $W[2]$. We extend their model of lobbying as mentioned above (see Section 2 for a formal description), and provide algorithms and analysis for these extended models in terms of classical and parameterized complexity. All complexity-theoretic notions needed will be presented in Section 3.

Our classical complexity results (presented in Section 4) are shown via either polynomial-time algorithms for or reductions showing NP-completeness of the problems studied. For the parameterized complexity results (presented in Section 5), we choose natural parameters such as The Lobby's budget, the budget per referendum, and the "discretization level" used in formalizing our probabilistic lobbying problems (see Section 2.1). We also consider the concept of issue weighting, modeling that certain issues will be of more importance to The Lobby than others. Our classical and parameterized complexity results are summarized in Table 1 (see Section 4) for problems without and in Table 2 (see Section 4.3) for problems with issue weighting.

In Section 6, we provide approximability and inapproximability results for probabilistic lobbying problems. In this way we add breadth and depth to not only the models but also the understanding of lobbying behavior. We conclude by summarizing our main results and stating some open problems in Section 7.

2. Models for probabilistic lobbying

2.1. Initial model

We begin with a version of the Probabilistic Lobbying Problem (PLP, for short) in which voters start with initial probabilities of voting for an issue and are assigned known costs for increasing their probabilities of voting according to

Table 2
Complexity results for X-Y-PLP-WIW, where $X \in \{\text{MB}, \text{VB}\}$ and $Y \in \{\text{SM}, \text{AM}\}$.

Problem	Classical complexity	Parameterized complexity			Stated in or implied by Thm./Cor.
		Total budget	Budget per issue	Total budget & discr. level	
MB-SM-PLP-WIW	NP-complete	FPT	?	(FPT)	4.4 and 5.6
MB-AM-PLP-WIW	NP-complete	FPT	?	(FPT)	4.4 and 5.6
VB-SM-PLP-WIW	NP-complete	FPT	$W[2]$-complete*	FPT	4.5, 5.7 and 5.8, 5.9
VB-AM-PLP-WIW	NP-complete	?	$W[2]$-hard	FPT	4.5, 5.7 and 5.8

"The Lobby's" agenda by each of a finite set of increments. The question, for this class of problems, is: Given the above information, along with an agenda and a fixed budget B, can The Lobby target its bribes in order to achieve its agenda?

The complexity of the problem seems to hinge on the evaluation criterion for what it means to "win a vote" or "achieve an agenda". We discuss the possible interpretations of evaluation and bribery later in this section.[2] First, however, we will formalize the problem by defining data objects needed to represent the problem instances. (A similar model was first discussed by Reinganum [6] in the continuous case and we translate it here to the discrete case. This will allow us to present algorithms for, and a complexity analysis of, the problem.)

Let $\mathbb{Q}_{[0,1]}^{m\times n}$ denote the set of $m \times n$ matrices over $\mathbb{Q}_{[0,1]}$ (the rational numbers in the interval $[0, 1]$). By slight abuse of notation,[3] we say $P \in \mathbb{Q}_{[0,1]}^{m\times n}$ is a *probability matrix (of size $m \times n$)*, where each entry $p_{i,j}$ of P gives the probability that voter v_i will vote "yes" for referendum (synonymously, for issue) r_j. The result of a vote is either a "yes" (represented by 1) or a "no" (represented by 0). Thus, we represent the result of any vote on all issues as a 0/1 vector $\vec{X} = (x_1, x_2, \ldots, x_n)$, which is sometimes also denoted as a string in $\{0, 1\}^n$.

We now associate with each voter/issue pair (v_i, r_j) a discrete price function $c_{i,j}$ for changing v_i's probability of voting "yes" for issue r_j. Intuitively, $c_{i,j}$ is a monotonic staircase function that gives the cost for The Lobby of raising or lowering (in discrete steps) the ith voter's probability of voting "yes" on the jth issue. A formal description is as follows.

Given the entries $p_{i,j} = a_{i,j}/b_{i,j}$ of a probability matrix $P \in \mathbb{Q}_{[0,1]}^{m\times n}$, where $a_{i,j} \in \mathbb{N} = \{0, 1, \ldots\}$ and $b_{i,j} \in \mathbb{N}_{>0} = \{1, 2, \ldots\}$, choose some $k \in \mathbb{N}$ such that $k + 1$ is a common multiple of all $b_{i,j}$, where $1 \le i \le m$ and $1 \le j \le n$, and partition the probability interval $[0, 1]$ into $k + 1$ steps of size $1/(k + 1)$ each.[4] The integer k will be called the *discretization level* of the problem instance, and each integer κ, $0 \le \kappa \le k + 1$ might be called a *(confidence) step*. For each $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$, $c_{i,j} : \{0, 1/(k+1), 2/(k+1), \ldots, k/(k+1), 1\} \to \mathbb{N}$ is the *(discrete) price function for $p_{i,j}$*, i.e., $c_{i,j}(\ell/(k+1))$ is the price for changing the probability of the ith voter voting "yes" on the jth issue from $p_{i,j}$ to $\ell/(k+1)$, where $0 \le \ell \le k+1$. Note that the domain of $c_{i,j}$ consists of $k + 2$ elements of $\mathbb{Q}_{[0,1]}$ including 0, $p_{i,j}$, and 1. In particular, we require $c_{i,j}(p_{i,j}) = 0$, i.e., a cost of zero is associated with leaving the initial probability of voter v_i voting on issue r_j unchanged. Note that $k = 0$ means $p_{i,j} \in \{0, 1\}$, i.e., in this case each voter either accepts or rejects each issue with certainty and The Lobby can only flip these results.[5] The image of $c_{i,j}$ consists of $k + 2$ nonnegative integers including 0 (the confidence steps), and we require that, for any two elements a, b in the domain of $c_{i,j}$, if $p_{i,j} \le a \le b$ or $p_{i,j} \ge a \ge b$, then $c_{i,j}(a) \le c_{i,j}(b)$. This guarantees monotonicity on the prices in both directions.

We represent the list of price functions associated with a probability matrix P as a table C_P, called *cost matrix* in the following, whose $m \cdot n$ rows give the price functions $c_{i,j}$ and whose $k + 2$ columns give the costs $c_{i,j}(\ell/(k + 1))$, where $0 \le \ell \le k + 1$. Occasionally, we use $c_{i,j}[\ell]$ to denote $c_{i,j}(\ell/(k+1))$, for $\ell \in \{0, 1, \ldots, k + 1\}$. Note that we choose the same k for each $c_{i,j}$, so we have the same number of columns in each row of C_P. The entries of C_P can be thought of as "price tags" indicating what The Lobby must pay in order to change the probabilities of voting.

The Lobby also has an integer-valued budget B and an "agenda", which we will denote as a vector $\vec{Z} \in \{0, 1\}^n$ for n issues, containing the outcomes The Lobby would like to see on these issues. For The Lobby, the prices for a bribe that moves the outcomes of a referendum into the wrong direction do not matter. Hence, if $\vec{Z}$ is zero at position j, then we set $c_{i,j}(a) = --$ (indicating an unimportant entry) for $a > p_{i,j}$, and if $\vec{Z}$ is one at position j, then we set $c_{i,j}(a) = --$ (indicating an unimportant entry) for $a < p_{i,j}$. Note that $c_{i,j}(a) = 0$ if and only if $a = p_{i,j}$.

For simplicity, we may assume that The Lobby's agenda is all "yes" votes, so the target vector is $\vec{Z} = 1^n$. This assumption can be made without loss of generality, since if there is a zero in $\vec{Z}$ at position j, we can flip this zero to one and also change the corresponding probabilities $p_{1,j}, p_{2,j}, \ldots, p_{m,j}$ in the jth column of P to $1 - p_{1,j}, 1 - p_{2,j}, \ldots, 1 - p_{m,j}$ (see the evaluation

[2] We stress that when we use the term "bribery" in this paper, it is meant in the sense of lobbying [2], not in the sense Faliszewski et al. [5] have in mind when defining bribery in elections (see also, e.g., [4,44,45]). In bribery in the context of lobbying one considers modifications to multiple independent issues, while in bribery in the context of elections one generally considers modifying rank orderings of interdependent candidates.

[3] The term "probability matrix" might be perceived as misleading, since neither the row entries nor the column entries of this matrix must add up to one. However, we nonetheless use this term to indicate that each matrix entry represents the acceptance probability of a voter for an issue.

[4] There is some arbitrariness in this choice of k. One might think of more flexible ways of partitioning $[0, 1]$. We have chosen this way for the sake of simplifying the representation, but we mention that all that matters is that for each i and j, the discrete price function $c_{i,j}$ is defined on the value $p_{i,j}$, and is set to zero for this value.

[5] This is the special case of OL [2].

criteria in Section 2.3 for how to determine the result of voting on a referendum). Moreover, the rows of the cost matrix C_P that correspond to issue j have to be mirrored.

Example 2.1. Consider the following problem instance with $k = 9$ (so there are $k+1 = 10$ steps), $m = 2$ voters, and $n = 3$ issues. We will use this as a running example for the rest of this paper. In addition to the above definitions for k, m, and n, we also give the following probability matrix P and cost matrix C_P for P. (Note that this example is normalized for an agenda of $\vec{Z} = 1^3$, which is why The Lobby has no incentive for lowering the acceptance probabilities, so those costs are omitted below.)

Our example consists of a probability matrix P:

	r_1	r_2	r_3
v_1	0.8	0.3	0.5
v_2	0.4	0.7	0.4

and the corresponding cost matrix C_P:

$c_{i,j}$	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$c_{1,1}$	--	--	--	--	--	--	--	--	0	100	140
$c_{1,2}$	--	--	--	0	10	70	100	140	310	520	600
$c_{1,3}$	--	--	--	--	--	0	15	25	70	90	150
$c_{2,1}$	--	--	--	--	0	30	40	70	120	200	270
$c_{2,2}$	--	--	--	--	--	--	--	0	10	40	90
$c_{2,3}$	--	--	--	--	0	70	90	100	180	300	450

In Section 2.2, we describe two bribery methods, i.e., two specific ways in which The Lobby can influence the voters. These will be referred to as *microbribery* (MB) and *voter bribery* (VB). In Section 2.3, we define two ways in which The Lobby can win a set of votes. These evaluation criteria will be referred to as *strict majority* (SM) and *average majority* (AM). The four basic probabilistic lobbying problems we will study (each a combination of MB/VB bribery under SM/AM evaluation) are defined in Section 2.4, and a modification of these basic problems with additional issue weighting is introduced in Section 2.5.

2.2. Bribery methods

We begin by first formalizing the bribery methods by which The Lobby can influence votes on issues. We will define two methods for donating this money.

2.2.1. Microbribery (MB)

The first method at the disposal of The Lobby is what we will call *microbribery*.[6] We define microbribery to be the editing of individual elements of the P matrix according to the costs in the C_P matrix. Thus The Lobby picks not only which voter to influence but also which issue to influence for that voter. This bribery method allows a very flexible version of bribery, and models private donations made to politicians or voters in support of specific issues.

More formally, if voter i is bribed with d dollars on issue j, then all entries $c_{i,j}[\ell]$, $0 \leq \ell \leq k+1$, of C_P are updated as follows:

$$c_{i,j}[\ell] := \begin{cases} -- & \text{if } (c_{i,j}[\ell] = --) \vee ((c_{i,j}[\ell] - d) \leq 0) \\ c_{i,j}[\ell] - d & \text{if } (c_{i,j}[\ell] - d) > 0. \end{cases}$$

Moreover, assuming The Lobby's target vector is 1^n, let $T = \text{argmax}\{\ell \mid c_{i,j}[\ell] = --\}$. Replace $c_{i,j}[T] := 0$ and update $p_{i,j} := T/(k+1)$.

Example 2.2 (*Continuing* Example 2.1)**.** To make this concrete, reconsidering our Example 2.1: Suppose we give \$100 to the second voter and ask her to change her opinion on the third issue. This would lead to the following update of C_P:

$c_{i,j}$	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$c_{1,1}$	--	--	--	--	--	--	--	--	0	100	140
$c_{1,2}$	--	--	--	0	10	70	100	140	310	520	600
$c_{1,3}$	--	--	--	--	--	0	15	25	70	90	150
$c_{2,1}$	--	--	--	--	0	30	40	70	120	200	270
$c_{2,2}$	--	--	--	--	--	--	--	0	10	40	90
$c_{2,3}$	--	--	--	--	--	--	--	0	80	200	350

[6] Although our notion was inspired by theirs, we stress that it should not be confused with the term "microbribery" used by Faliszewski et al. [4,44,45] in the different context of bribing "irrational" voters in Llull/Copeland elections via flipping single entries in their preference tables.

Accordingly, the matrix P is updated as follows:

	r_1	r_2	r_3
v_1	0.8	0.3	0.5
v_2	0.4	0.7	0.7

2.2.2. Voter bribery (VB)

The second method at the disposal of The Lobby is *voter bribery*. We can see from the P matrix that each row represents what an individual voter thinks about all the issues on the docket. In this method of bribery, The Lobby picks a voter and then pays to edit the entire row at once with the funds being equally distributed over all the issues.[7] So, for d dollars a fraction of d/n is spent on each issue, and the probabilities change accordingly. The cost of moving the voter is given by the C_P matrix as before. This method of bribery is analogous to "buying" or pushing a single politician or voter. The Lobby seeks to donate so much money to some individual voters that they have no choice but to move all of their votes toward The Lobby's agenda.

Let us be more precise. To avoid problems with fractions of dollars, we will assume that the bribery money is donated in multiples of n, the number of issues. Hence, whole dollars will be donated per referendum. So, if we bribe voter i by giving her $x \cdot n$ dollars, this results in microbribing every issue by giving x dollars to voter i to raise her confidence on issue j; in other words, all the entries of $c_{i,j}$ (in C_P), $1 \leq j \leq n$, will be edited (and accordingly P).

Example 2.3 (*Continuing* Example 2.1)**.** Let us reconsider our running example: What happens if we give \$300 to the second voter? The update of C_P would be as follows:

$c_{i,j}$	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$c_{1,1}$	--	--	--	--	--	--	--	--	0	100	140
$c_{1,2}$	--	--	--	0	10	70	100	140	310	520	600
$c_{1,3}$	--	--	--	--	--	0	15	25	70	90	150
$c_{2,1}$	--	--	--	--	--	--	--	0	20	100	170
$c_{2,2}$	--	--	--	--	--	--	--	--	--	--	0
$c_{2,3}$	--	--	--	--	--	--	--	0	80	200	350

Accordingly, the matrix P is updated as follows:

	r_1	r_2	r_3
v_1	0.8	0.3	0.5
v_2	0.7	1.0	0.7

Note that microbribery is equivalent to voter bribery if there is only one referendum.

2.3. Evaluation criteria

Defining criteria for how an issue is won is the next important step in formalizing our models. Here we define two methods that one could use to evaluate the eventual outcome of a vote. Since we are focusing on problems that are probabilistic in nature, it is important to note that no evaluation criterion will guarantee a win. The criteria below yield different outcomes depending on the model and problem instance.

2.3.1. Strict majority (SM)

For each issue, a strict majority of the individual voters have probability greater than some threshold, t, of voting according to the agenda. In our running example (see Example 2.1), with $t = 50\%$, the result of the votes would be $\vec{X} = (0, 0, 0)$, because none of the issues has a strict majority of voters with above 50% likelihood of voting "yes". The bribery action described in Example 2.2 results in the same vector, $(0, 0, 0)$. However, the bribery action described in Example 2.3 results in the vector $(1, 0, 0)$.

2.3.2. Average majority (AM)

For each issue r_j of a given probability matrix P, we define the average probability $\overline{p_j} = \left(\sum_{i=1}^{m} p_{i,j}\right)/m$ of voting "yes" for r_j. We now evaluate the vote to say that r_j is accepted if and only if $\overline{p_j} > t$ where t is some threshold. In our running example

[7] Note that at first glance it might seem more appealing to allow the voters to specify what percentage of each payment is spent on which issue. However, this is in fact not needed: The same effect can be achieved by scaling the prices in the cost matrix appropriately.

with $t = 50\%$, this would give us a result vector of $\vec{X} = (1, 0, 0)$. However, the bribery action described in Example 2.2 results in the vector (1, 0, 1), while the bribery action described in Example 2.3 results in the vector (1, 1, 1).

Note that the first two criteria coincide if there is only one voter or if the discretization level equals zero.

A natural evaluation criteria would be to take into account the probabilities of possible *scenarios* or futures. Each possible future, where voters commit to "yes" or "no" votes, has a probability of occurring. If we sum the probabilities of the positive future scenarios, we would know the exact probability of a successful outcome. This model has been studied in another paper [46] and led to very complex models. In this paper we have chosen to analyze two proxy models instead, to see if relaxing the evaluation criteria leads to easy computational problems.

2.4. Basic probabilistic lobbying problems

We now introduce the four basic problems that we will study. Recalling that, without loss of generality, The Lobby's target vector may be assumed to be all ones, we define the following problem for $X \in \{\text{MB}, \text{VB}\}$ and $Y \in \{\text{SM}, \text{AM}\}$.

	X-Y Probabilistic Lobbying Problem
Given:	A probability matrix $P \in \mathbb{Q}_{[0,1]}^{m \times n}$ with a cost matrix C_P (with integer entries), a budget B, and some threshold $t \in \mathbb{Q}_{[0,1]}$.
Question:	Is there a way for The Lobby to influence C_P and hence P (using bribery method X and evaluation criterion Y, without exceeding budget B) such that the result of the votes on all issues equals 1^n?

We abbreviate this problem name as X-Y-PLP. Recall that C_P has $(k + 2)mn$ many entries.

Example 2.4 (*Continuing* Example 2.1)**.** Consider voter bribery and the average majority criterion with our running example and suppose The Lobby has a budget of \$75, i.e., our instance of VB-AM-PLP is $(P, C_P, 75)$ with P and C_P as given in Example 2.1. Giving \$75 to the first voter would suffice to lift all issues above the threshold of 50% on average according to the wishes of The Lobby. The updated cost matrix C_P' would be:

$c_{i,j}$	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$c_{1,1}$	--	--	--	--	--	--	--	--	0	75	115
$c_{1,2}$	--	--	--	--	0	45	75	115	285	495	575
$c_{1,3}$	--	--	--	--	--	--	--	0	45	65	125
$c_{2,1}$	--	--	--	--	0	30	40	70	120	200	270
$c_{2,2}$	--	--	--	--	--	--	--	0	10	40	90
$c_{2,3}$	--	--	--	--	0	70	90	100	180	300	450

This leads to the following updated probability matrix P', enriched with the average probabilities:

	r_1	r_2	r_3
v_1	0.8	0.4	0.7
v_2	0.4	0.7	0.4
$\overline{p_j}$	0.6	0.55	0.55

Since each referendum passes the evaluation test, as desired by The Lobby, $(P, C_P, 75)$ is in VB-AM-PLP.

Notice that the discretization level is an implicit (unary) parameter of the problem that is indirectly specified through the cost matrix C_P.

2.5. Probabilistic lobbying with issue weighting

We now augment the model to include the concept of issue weighting. It is reasonable to surmise that certain issues will be of more importance to The Lobby than others. For this reason we will allow The Lobby to specify higher weights to the issues that they deem more important. These positive integer weights will be defined for each issue.

We will specify these weights as a vector $\vec{W} \in \mathbb{N}_{>0}^n$ with size n equal to the total number of issues in our problem instance. The higher the weight, the more important that particular issue is to The Lobby. Along with the weights for each issue we are also given an objective value $O \in \mathbb{N}_{>0}$, which is the minimum weight The Lobby wants to see passed. Since this is a partial ordering, it is possible for The Lobby to have an ordering such as $w_1 = w_2 = \cdots = w_n$. If this is the case, we see that we are left with an instance of X-Y-PLP, where $X \in \{\text{MB}, \text{VB}\}$ and $Y \in \{\text{SM}, \text{AM}\}$.

We now introduce the four probabilistic lobbying problems with issue weighting. For $X \in \{\text{MB}, \text{VB}\}$ and $Y \in \{\text{SM}, \text{AM}\}$, we define the following problem.

	X-Y Probabilistic Lobbying Problem with Issue Weighting
Given:	A probability matrix $P \in \mathbb{Q}^{m \times n}_{[0,1]}$ with cost matrix C_P, an issue weight vector $\vec{W} \in \mathbb{N}^n_{>0}$, an objective value $O \in \mathbb{N}_{>0}$, a budget B, and some threshold $t \in \mathbb{Q}_{[0,1[}$.
Question:	Is there a way for The Lobby to influence C_P and hence P (using bribery method X and evaluation criterion Y, without exceeding budget B) such that the total weight of all issues for which the result coincides with The Lobby's target vector 1^n is at least O?

We abbreviate this problem name as X-Y-PLP-WIW.

3. Complexity-theoretic notions

We assume the reader is familiar with standard notions of (classical) complexity theory, such as P, NP, and NP-completeness. Since we analyze the problems stated in Section 2 not only in terms of their classical complexity, but also with regard to their *parameterized* complexity, we provide some basic notions here (see, e.g., the text books by Downey and Fellows [47], Flum and Grohe [48], and Niedermeier [49] for more background). As we derive our results in a rather specific fashion, we will employ the "Turing way" as proposed by Cesati [50]. The reader familiar with these notions can immediately proceed to the next section.

A *parameterized problem* $\mathscr{P}$ is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is a fixed alphabet and Σ^* is the set of strings over Σ. Each instance of the parameterized problem $\mathscr{P}$ is a pair (I, p), where the second component p is called the *parameter*. The language $L(\mathscr{P})$ is the set of all YES instances of $\mathscr{P}$. The parameterized problem $\mathscr{P}$ is *fixed-parameter tractable* if there is an algorithm (realizable by a deterministic Turing machine) that decides whether an input (I, p) is a member of $L(\mathscr{P})$ in time $f(p)|I|^c$, where c is a fixed constant and f is a function of the parameter p, but is independent of the overall input length, $|I|$. The class of all fixed-parameter tractable problems is denoted by FPT.

Sometimes, more than one parameter (e.g., two parameters (p_1, p_2)) are associated with a (classical) problem. This is formally captured in the definition above by coding those parameters into one number p via a so-called pairing function through diagonalization. As is standard, we assume our pairing function to be a polynomial-time computable bijection from $\mathbb{N} \times \mathbb{N}$ onto $\mathbb{N}$ that has polynomial-time computable inverses.

For a given classical decision or optimization problem, there are various ways to define parameters. With minimization problems, the *standard parameterization* is a bound on the entity to be minimized. For instance, the problems studied in this paper have, as a natural minimization objective, the goal to minimize costs (i.e., to use a budget B as small as possible). If one can assume that the parameter p is small in practice, or in practical situations involving humans, we can argue that an algorithm offering a running time of $\mathcal{O}(2^p|x|)$ for an instance x with parameter p behaves reasonably well in practice. Admittedly, this might not be the case with B itself, in particular in view of the fact that only $\log(B)$ bits affect the overall size of the instance. However, this is still the parameter one should start from when considering minimization problems in order to classify them from the perspective of parameterized complexity.

Are there (more) reasonable, i.e., smaller parameters worth investigating? A related natural parameter choice would be the budget that can be spent per issue, i.e., the entity B/n. If the voters are actual human beings, one can also argue that the discretization level k would not be too large.

One of the current trends in parameterized complexity analysis is to study multiple parameterizations for each problem, including combining multiple parameters for a problem instance. This trend is highlighted by two recent invited talks given by Fellows [51] and Niedermeier [52]. Notice that the study of different and multiple parameterizations can also be seen from another angle: Apart from identifying the hard parts of the problem instance, such research represents a natural mathematical counterpart of the more practically oriented quest for good parameters that may lead to the most competitive algorithm frameworks for hard problems, as exemplified most notably by SATzilla and ParamILS in the areas of algorithms for Satisfiability and Integer Linear Programming, respectively, see [53,54].

There is also a theory of parameterized complexity, as exhibited in [47–49], where parameterized complexity is expressed via hardness for or completeness in the levels $W[t]$, $t \geq 1$, of the W-hierarchy, which includes fixed-parameter tractability at its lower end and is built on top of it to characterize, level by level, higher degrees of parameterized intractability:

$$\text{FPT} = W[0] \subseteq W[1] \subseteq W[2] \subseteq \cdots.$$

It is commonly believed that this hierarchy is strict. Since only the second level, $W[2]$, will be of interest to us in this paper, we will define only this class below.

Definition 3.1. Let $\mathscr{P}$ and $\mathscr{P}'$ be two parameterized problems. A *parameterized reduction from* $\mathscr{P}$ *to* $\mathscr{P}'$ is a function r that, for some polynomial q and some function g, is computable in time $\mathcal{O}(g(p)q(|I|))$ and maps an instance (I, p) of $\mathscr{P}$ to an instance $r(I, p) = (I', p')$ of $\mathscr{P}'$ such that

1. (I, p) is a YES instance of $\mathscr{P}$ if and only if (I', p') is a YES instance of $\mathscr{P}'$, and
2. $p' \leq g(p)$.

We then say that $\mathscr{P}$ *parameterized reduces to* $\mathscr{P}'$ (*via* r). Parameterized hardness for and completeness in a parameterized complexity class is defined via parameterized reductions. We will show only $W[2]$-completeness results. A parameterized problem $\mathscr{P}'$ is said to be $W[2]$-*hard* if every parameterized problem $\mathscr{P}$ in $W[2]$ parameterized reduces to $\mathscr{P}'$. $\mathscr{P}'$ is said to be $W[2]$-*complete* if $\mathscr{P}'$ is in $W[2]$ and is $W[2]$-hard.

Notice that one can find at least two other competing definitions of parameterized reduction in the literature: the second condition, $p' \leq g(p)$, is sometimes restricted to $p' \leq p$ and sometimes only requires $p' \leq f(p)$ for some function f that need not coincide with the function g occurring in the running time bound. In particular, the second variant is easily seen to be equivalent to our formulation, as $g'(x) = \max\{g(x), f(x)\}$ shows that a reduction being computable in time $\mathcal{O}(g(p)q(|I|))$ and also satisfying $p' \leq f(p)$ is clearly computable in time $\mathcal{O}(g'(p)q(|I|))$ and verifies $p' \leq g'(p)$.

$W[2]$ can be characterized by the following problem on Turing machines:

	Short Multi-tape Nondeterministic Turing Machine Computation
Given:	A multi-tape nondeterministic Turing machine M (with two-way infinite tapes) and an input string x (both M and x are given in some standard encoding).
Parameter:	A positive integer k.
Question:	Is there an accepting computation of M on input x that reaches a final accepting state in at most k steps?

Note that the complexity of Short Multi-tape Nondeterministic Turing Machine Computation crucially depends on the choice of parameters [47]. It is also important that the machine accesses its tapes in parallel.

More specifically, a parameterized problem $\mathscr{P}$ is in $W[2]$ if and only if it can be reduced to Short Multi-tape Nondeterministic Turing Machine Computation via a parameterized reduction [50]. This can be accomplished by giving an appropriate multi-tape nondeterministic Turing machine for solving $\mathscr{P}$. Hardness for $W[2]$ can be shown by giving a parameterized reduction in the opposite direction, from Short Multi-tape Nondeterministic Turing Machine Computation to $\mathscr{P}$.

For other applications of fixed-parameter tractability and parameterized complexity to problems from computational social choice, see, e.g., [55].

4. Classical complexity results

We now provide a formal complexity analysis of the probabilistic lobbying problems for all combinations of bribery methods $X \in \{\text{MB}, \text{VB}\}$ and evaluation criteria $Y \in \{\text{SM}, \text{AM}\}$. Table 1 summarizes some of our (classical and parameterized) complexity results for the problems X-Y-PLP; note that Table 1 does not cover Theorem 5.5 (which considers a combination of two parameters, namely of budget per issue and discretization level).

Some of these results are known from previous work by Christian et al. [2], as will be mentioned below. Our results generalize theirs by extending the model to probabilistic settings. The listed FPT results might look peculiar at first glance, since Christian et al. [2] derived $W[2]$-hardness results, but this is due to the chosen parameterization, as will be discussed later in more detail. We put parentheses around some classes in Table 1 to indicate that these results are trivially inherited from others. For example, if some problem is solvable in polynomial time, then it is in FPT for any parameterization. The table mainly provides results on the containment of problems in certain complexity classes; if known, additional hardness results are also listed.

In Section 4.1 we present our results on microbribery (i.e., we study the problems MB-Y-PLP for $Y \in \{\text{SM}, \text{AM}\}$), and in Section 4.2 we are concerned with voter bribery (i.e., we study the problems VB-Y-PLP for $Y \in \{\text{SM}, \text{AM}\}$). In addition, in Section 4.3 we study probabilistic lobbying with issue weighting.

4.1. Microbribery

Theorem 4.1. MB-SM-PLP *is in P.*

Proof. The aim is to win all referenda. For each voter v_i and referendum r_j, $1 \leq i \leq m$ and $1 \leq j \leq n$, we compute in polynomial time the amount $b(v_i, r_j)$ The Lobby has to spend to turn the favor of v_i in the direction of The Lobby (beyond the given threshold t). In particular, set $b(v_i, r_j) = 0$ if voter v_i would already vote according to the agenda of The Lobby. For each issue r_j, sort $\{b(v_i, r_j) \mid 1 \leq i \leq m\}$ nondecreasingly, yielding a sequence $b_1(r_j), \ldots, b_m(r_j)$ such that $b_k(r_j) \leq b_\ell(r_j)$ for $k < \ell$. To win referendum r_j, The Lobby must spend at least $B(r_j) = \sum_{i=1}^{\lceil (m+1)/2 \rceil} b_i(r_j)$ dollars. Hence, all referenda are won if and only if $\sum_{j=1}^{n} B(r_j)$ is at most the given bribery budget B. □

Note that the time needed to implement the algorithm given in the previous proof can be bounded by a polynomial of low order. More precisely, if the input consists of m voters, n referenda, and discretization level k, then $\mathcal{O}(n \cdot m \cdot k)$ time is needed to compute the $b(v_i, r_j)$. Having these values, $\mathcal{O}(n \cdot m \cdot \log m)$ time is needed for the sorting phase. The sums can be computed in time $\mathcal{O}(n \cdot m)$. (Note that the time analysis can still be improved; however, a rough estimate of the computation time needed is enough to establish Theorem 4.1.)

Similarly, the other problems that we show to belong to P admit solution algorithms bounded by polynomials of low order.

Theorem 4.2. MB-AM-PLP *is in* P.

Proof. Let (P, C_P, B, t) be a given MB-AM-PLP instance, where $P \in \mathbb{Q}_{[0,1]}^{m\times n}$, C_P is a cost matrix, B is The Lobby's budget, and t is a given threshold. Let k be the discretization level of P, i.e., the interval is divided into $k+1$ steps of size $1/(k+1)$ each. For $j \in \{1, 2, \ldots, n\}$, let d_j be the minimum cost for The Lobby to bring referendum r_j into line with the jth entry of its target vector 1^n. If $\sum_{j=1}^{n} d_j \leq B$, then The Lobby can achieve its goal that the votes on all issues pass.

We show that for a fixed j we can compute d_j in polynomial time. Therefore, the decision problem of whether The Lobby can afford to bring all referenda into line with its target vector is also in P. We compute d_j by dynamic programming. The aim is to have $\sum_{i=1}^{m} p_{i,j}/m > t$, i.e., $\sum_{i=1}^{m} p_{i,j} > mt$. Recall that $p_{i,j} \cdot (k+1)$ always gives an integer. Define $c_j = (k+1)(mt - \sum_{i=1}^{m} p_{i,j}) + 1$. This is the overall number of confidence steps The Lobby has to buy to win referendum r_j. Note that c_j is polynomial in the size of the input, as is mtk. We define $T[l, s]$ to be the minimum cost of raising $(k+1)(\sum_{i=1}^{m} p_{i,j})$ by value $s \leq c_j$ using only microbribes to the first l voters. Notice that $T[l, 0] = 0$. Let $q_{i,j}$ be the integer with $c_{i,j}[q_{i,j}] = 0$. Hence, for κ with $0 \leq \kappa < q_{i,j}$, we find $c_{i,j}[\kappa] = -\!-$, and for κ with $q_{i,j} < \kappa \leq k+1$, we have $c_{i,j}[\kappa] > 0$. This means that $q_{i,j}/(k+1) = p_{i,j}$. As the maximum number of confidence steps we can gain by bribing voter i is $k + 1 - q_{i,j}$, we obtain

$$T[1, s] = \begin{cases} \infty & \text{if } s > k + 1 - q_{1,j}, \\ c_{1,j}[q_{1,j} + s] & \text{otherwise.} \end{cases}$$

Based on this initialization, we can compute:

$$T[l, s] = \min\left\{T[l-1, s-q] + c_{l,j}[q_{l,j} + q] \mid 0 \leq q \leq \min\{s, k+1-q_{l,j}\}\right\}.$$

In particular, $q = 0$ covers the case when no money is spent on voter l, as $c_{l,j}[q_{l,j}] = 0$. Thus, each of the polynomially many entries, $T[l, s]$, can be computed in polynomial time. In particular, $T[m, c_j] = d_j$ can be computed in polynomial time. □

4.2. Voter bribery

Recall the Optimal Lobbying problem (OL) defined in Section 1.1. Again, The Lobby's target vector $\vec{Z}$ may be assumed to be all ones, without loss of generality, so $\vec{Z}$ may be dropped from the input.

Christian et al. [2] proved that this problem is $W[2]$-complete by reducing from the $W[2]$-complete problem k-Dominating Set to OL (showing the lower bound) and from OL to the $W[2]$-complete problem Independent-k-Dominating Set (showing the upper bound). In particular, this implies NP-hardness of OL.

The following result focuses on the classical complexity of VB-SM-PLP and VB-AM-PLP; the parameterized complexity of these problems will be studied in Section 5 and will make use of the proof of Theorem 4.3 below.

To employ the $W[2]$-hardness result of Christian et al. [2], we show that OL is a special case of VB-SM-PLP and thus (parameterized) polynomial-time reduces to VB-SM-PLP. Analogous arguments apply to VB-AM-PLP.

Theorem 4.3. VB-SM-PLP *and* VB-AM-PLP *are* NP*-complete.*

Proof. Membership in NP is obtained through a "guess-and-check" algorithm for VB-SM-PLP and VB-AM-PLP.

We now prove that VB-SM-PLP is NP-hard by reducing OL to VB-SM-PLP. We are given an instance (E, b) of OL, where E is a $m \times n$ 0/1 matrix and b is the number of votes to be edited. Recall that The Lobby's target vector is 1^n. We construct an instance of VB-SM-PLP consisting of the given matrix $P = E$ (a "degenerate" probability matrix with only the probabilities 0 and 1), a corresponding cost matrix C_P, and a budget B. C_P has two columns (we have $k = 0$, since the problem instance is deterministic, see Section 2.1), one column for probability 0 and one for probability 1. All entries of C_P corresponding to $p_{i,j} \neq 1$ are set to unit cost: $c_{i,j}[1] = 1$ if $p_{i,j} \neq 1$. The threshold t is set to 1/2.

The cost of increasing any value in P is n, since donations are distributed evenly across issues for a given voter. We want to know whether there is a set of bribes of cost at most $b \cdot n = B$ such that The Lobby's agenda passes. This holds if and only if there are b voters that can be bribed so that they vote uniformly according to The Lobby's agenda and that is sufficient to pass all the issues. Thus, the given instance (E, b) is in OL if and only if the constructed instance (P, C_P, B, t) is in VB-SM-PLP, which shows that OL is a polynomial-time recognizable special case of VB-SM-PLP, and thus VB-SM-PLP is NP-hard.

Note that for the construction above it does not matter whether we use the strict-majority criterion (SM) or the average-majority criterion (AM). Since the entries of P are 0 or 1, we have $\overline{p_j} > 0.5$ if and only if we have a strict majority of ones in the jth column. Thus, VB-AM-PLP is NP-hard, too. □

4.3. Probabilistic lobbying with issue weighting

Table 2 summarizes some of our results for X-Y-PLP-WIW, where $X \in \{\text{MB}, \text{VB}\}$ and $Y \in \{\text{SM}, \text{AM}\}$; again, note that Table 2 does not cover all our results. The most interesting observation from the table is that introducing issue weights

raises the complexity from P to NP-completeness for all cases of microbribery (though it remains the same for voter bribery). Nonetheless, we show (Theorem 5.6) that these NP-complete problems are fixed-parameter tractable. Another interesting observation concerns the question of membership in $W[2]$. In the case indicated by the $*$ annotation, we can show this membership only when we take the lower bound O quantifying the objective of the bribery (in terms of issue weights) as a further parameter. Question marks indicate open problems.

Theorem 4.4. MB-SM-PLP-WIW *and* MB-AM-PLP-WIW *are each* NP*-complete.*

Proof. Membership in NP can be shown with a "guess-and-check" algorithm for both problems. To prove that MB-SM-PLP-WIW is NP-hard, we give a reduction from the well-known NP-complete problem Knapsack (see, e.g., [56]) to the problem MB-SM-PLP-WIW. In Knapsack, we are given a set of objects $U = \{o_1, \ldots, o_n\}$ with weights $w : U \to \mathbb{N}$ and profits $p : U \to \mathbb{N}$, and $W, P \in \mathbb{N}$. The question is whether there is a subset $J \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in J} w(o_i) \leq W$ and $\sum_{i \in J} p(o_i) \geq P$. Given a Knapsack instance (U, w, p, W, P), create an MB-SM-PLP-WIW instance with $k = 0$ and only one voter, v_1, where for each issue, v_1's acceptance probability is either zero or one. For each object $o_j \in U$, create an issue r_j such that the acceptance probability of v_1 is zero. Let the cost of raising this probability on r_j be $c_{1,j}(1) = w(o_j)$ and let the weight of issue r_j be $w_j = p(o_j)$. Let The Lobby's budget be W and its objective value be $O = P$. Set the threshold t to $1/2$. By construction, there is a subset $J \subseteq \{1, \ldots, n\}$ with $\sum_{i \in J} w(o_i) \leq W$ and $\sum_{i \in J} p(o_i) \geq P$ if and only if there is a subset $J \subseteq \{1, \ldots, n\}$ with $\sum_{i \in J} c_{1,i}(1) \leq W$ and $\sum_{i \in J} w_i \geq O$.

As the reduction introduces only one voter, there is no difference between the evaluation criteria SM and AM. Hence, the above reduction works for both problems. □

Turning now to voter bribery with issue weighting, note that an immediate consequence of Theorem 4.3 is that VB-SM-PLP-WIW and VB-AM-PLP-WIW are NP-hard, since they are generalizations of VB-SM-PLP and VB-AM-PLP, respectively. Again, membership in NP can be seen using appropriate "guess-and-check" algorithms for the more general problems.

Corollary 4.5. VB-SM-PLP-WIW *and* VB-AM-PLP-WIW *each are* NP*-complete.*

NP-completeness of our problems with issue weighting implies that bribery/lobbying is hard in these settings, at least in the classical sense of worst-case complexity. However, in Section 5 we will provide some positive worst-case results for these problems in terms of their fixed-parameter tractability, and in Section 6 we will show that many of these problems admit good approximation schemes and may therefore be easy to solve in practice.

5. Parameterized complexity results

In this section, we study the parameterized complexity of our probabilistic lobbying problems. Parameterized hardness is usually shown by proving hardness for the levels of the W-hierarchy (with respect to parameterized reductions). Indeed, this hierarchy may be viewed as a "barometer of parametric intractability" [47, p. 14]. The lowest two levels of the W-hierarchy, $W[0] = \text{FPT}$ and $W[1]$, are the parameterized analogs of the classical complexity classes P and NP. We will show completeness results for the $W[2]$ level of this hierarchy.

In parameterized complexity, the standard parameterization for minimization problems is an upper bound on the entity to be minimized. In our case, this is the budget B. Since in the voter bribery model, the money is equally distributed over all referenda, it also makes sense to consider the upper bound B/n, i.e., the budget per referendum, as a natural, derived parameter for that scenario. A different, more mathematical reason for considering this parameter is the fact that, first of all, with the parameter B alone, most problems will find their home in the lowest complexity class, FPT, so it is reasonable to consider parameters that are distinctively smaller than B, which clearly applies to B/n. In particular, while B might be quite a big number, B/n can be assumed to be small in practical circumstances. Another natural way of parameterization is derived from certain properties of the input, be they implicit or explicit. In our case, the discretization level can be considered as such a parameter, in particular, since the smallest discretization level has been already considered before within the Optimal Lobbying problem [2]. Therefore, we examine all three of these parameterizations in order to understand the effect the choice of parameterizations has on the complexity of the problems.

5.1. Voter bribery

Theorem 5.1. VB-SM-PLP *and* VB-AM-PLP *parameterized by the budget and by the discretization level are in* FPT.

Proof. Consider an instance of VB-Y-PLP, $Y \in \{\text{SM}, \text{AM}\}$, i.e., we are given n referenda and m voters, as well as a cost matrix C_P (with either $--$ or integer entries), a discretization level k, a budget B, and a threshold t. Recall that the target vector $\vec{Z}$ of The Lobby is assumed to be 1^n. By the definition of our staircase price functions, the rows of C_P are monotonically nondecreasing (after possibly some $--$ entries). Observe that any successful bribe of any voter needs at least n dollars, since the money is evenly distributed among all referenda, and at least one dollar is needed to influence the chosen voter's votes for all referenda. Hence, $B \geq n$. We can assume that any entry in C_P is limited by $B + 1$, after replacing every entry bigger than B by $B + 1$. Notice that the entry $B + 1$ reflects that the intended bribery cannot be afforded.

Although k could be bigger than B, the interesting area of each row in C_P (containing integer entries) cannot have more than B strict increases in the sequence. We therefore encode each row in C_P by a sequence $(k_1, b_1, k_2, b_2, \ldots, k_\ell, b_\ell)$, $\ell \leq B$, which reads as follows: By investing b_j dollars, we proceed to column number $\sum_{i \leq j} k_i$. Note that k is given in unary in the original instance (implicitly by giving the cost matrix C_P), and that each k_j can be encoded with $\log k$ bits. Hence, we extract from C_P for each voter v a submatrix $S_P(v)$ with $n \leq B$ rows (for the referenda) and at most $2B$ columns (encoding the "jumps" in the integer sequence as described above). This matrix with at most B rows and at most $2B$ columns can be alternatively viewed as a matrix with at most B rows and at most B columns, where each matrix entry consists of a pair of numbers, one between 1 and $B+1$ and one being at most k. Therefore, we can associate with each voter at most $((B+1) \cdot k)^{B^2}$ distinct submatrices $S_P(v)$ of this kind, called voter profiles. It makes no sense to store more than B voters with the same profile. Hence, we can assume that $m \leq B \cdot ((B+1) \cdot k)^{B^2}$. Therefore, all relevant parts of the input are bounded by a function in the parameters B and k,[8] so that a brute-force algorithm can be used to solve the instance. This shows that the problem is in FPT. □

If we assume that the discretization level is a rather small number, the preceding theorem says that the problems VB-SM-PLP and VB-AM-PLP can be solved efficiently in practice. Although we were not able to establish an FPT result for VB-AM-PLP when the discretization level is not part of the parameter (but only the budget is), we can overcome this formal obstacle for VB-SM-PLP, as the following result shows.

Theorem 5.2. VB-SM-PLP *parameterized by the budget is in* FPT.

Proof. Let an instance of VB-SM-PLP be given. From the given cost matrix C_P, we extract the information $W(i, j)$ that gives the minimum amount of money The Lobby must spend on voter v_i to turn his or her voting behavior on issue r_j in favor of The Lobby's agenda, eventually raising the corresponding voting probability beyond the given threshold t. Each entry in $W(i, j)$ is between 0 and B. Moreover, as argued in the previous proof, there are no more than B issues and we again define a voter profile (this time the ith row of the table $W(i, j)$ gives such a profile) for each voter, and we need to keep at most B voters with the same profile. Hence, no more than $B(B+1)^B$ voters are present in the instance. Therefore, some brute-force approach can be used to show membership in FPT. □

The area of parameterized complexity leaves some freedom regarding the choice of parameterization. The main reason that the standard parameterization (referring to the entity to be minimized, in this case the budget) yields an FPT result is the fact that the parameter is already very big compared to the overall input (e.g., the number of issues n) by the very definition of the problem: Since the money given to one voter will be evenly distributed among the issues and since the cost matrix contains only integer entries, it makes no sense at all to spend less than n dollars on a voter. Hence, the budget should be at least n dollars (assuming that some of the voters must be influenced by The Lobby to achieve their agenda). This obstacle can be sidestepped by changing the parameterization to B/n, i.e., to the "budget per issue" (see, e.g., Theorem 5.3). Note that another way would be allowing rational numbers as entries in the cost matrix but we will not consider this in this paper but rather focus on the "budget per issue".

Theorem 5.3. VB-SM-PLP *parameterized by the budget per issue is* $W[2]$*-complete.*

Proof. $W[2]$-hardness can be derived from the proof of Theorem 4.3. Recall that in the proof of this theorem an instance (E, b) of OL was reduced to an instance of VB-SM-PLP, with budget $B = n \cdot b$. Hence, the parameter "budget per issue" of that VB-SM-PLP instance equals b. Therefore, the reduction in the proof of Theorem 4.3 preserves the parameter and hence $W[2]$-hardness follows from the $W[2]$-hardness of OL, see [2]. Moreover, the instance of VB-SM-PLP produced by the reduction has discretization level zero.

To show membership in $W[2]$, we reduce VB-SM-PLP to Short Multi-tape Nondeterministic Turing Machine Computation, which was defined in Section 3. To this end, it suffices to describe how a nondeterministic multi-tape Turing machine can solve such a lobbying problem.

Consider an instance of VB-SM-PLP: a probability matrix $P \in \mathbb{Q}_{[0,1]}^{m \times n}$ with a cost matrix C_P, a budget B, and a fixed threshold t. We may identify t with a certain step level for the price functions.

The reducing machine works as follows. From P, C_P, and t, the machine extracts the information $H_{i,j}(d)$, $1 \leq d \leq B$, where $H_{i,j}(d)$ is true if $p_{i,j} \geq t$ or if $c_{i,j}(t) \leq d/n$. Note that the bribery money is evenly distributed across all issues, also note that $H_{i,j}(d)$ captures whether paying d dollars to voter v_i helps to raise the acceptance probability of v_i on referendum r_j above the threshold t. Moreover, for each referendum r_j, the reducing machine computes the minimum number of voters that need to switch their opinion so that majority is reached for that specific referendum; let $s(j)$ denote this threshold for r_j. Since the cost matrix contains integer entries, meaningfully bribing s voters costs at least $s \cdot n$ dollars; only then each referendum will receive at least one dollar per voter. Hence, a referendum with $s(j) > B/n$ yields a NO instance. We can therefore replace any value $s(j) > B/n$ by the value $\lfloor B/n \rfloor + 1$.

From $H_{i,j}(d)$, the reducing machine produces (basically by sorting) another *winning table* $W_i(\ell)$ that lists for voter v_i those referenda where the acceptance probability of v_i on referendum r_j is raised above the threshold t by paying to v_i the amount

[8] In technical terms, this means that we have derived a so-called problem kernel for this problem.

of $\ell \cdot n$ dollars but not by paying $(\ell - 1) \cdot n$ dollars. Note that we can assume that the bribery money is spent in multiples of n, the number of referenda, since spending n dollars on some voter means spending one dollar per issue for that voter. This table is initialized by $W_i(0)$ listing those referenda already won at the very beginning, although this is not an important issue due to the information contained in $s(j)$.

The nondeterministic multi-tape Turing machine M we describe next has, in particular, access to $W_i(\ell)$ and to $s(j)$. M has $n + 1$ working tapes T_j, $0 \leq j \leq n$, all except one of which correspond to issues r_j, $1 \leq j \leq n$. We will use the set of voters, $V = \{v_1, \ldots, v_m\}$, as part of the work alphabet. The (formal) input tape of M is ignored.

M starts by writing $s(j)$ symbols # onto tape j for each j, $1 \leq j \leq n$. By using simultaneous writing steps, this needs at most $\lfloor B/n \rfloor + 1$ steps, since $s(j) \leq \lfloor B/n \rfloor + 1$ as argued above. We also need an "information hiding" trick here: Every time the machine writes a # symbol, it simultaneously moves the writing head one field to the right, so that in the next step the head will read a blank symbol. The trick is required in order to keep the transition table small: basically, we cannot insert in the transition table 2^n different instructions to take into account all different configurations of blank and # symbols on the n tapes.

Second, for each $i \in \{1, \ldots, m\}$, M writes k_i symbols v_i from the alphabet V on the zeroth tape, T_0, such that $\sum_{i=1}^{m} k_i \leq B/n$. This is the nondeterministic guessing phase where the amount of bribery money spent on each voter, namely $k_i \cdot n$ for voter v_i, is determined. The finite control is used to ensure that a word from the language $\{v_1\}^* \cdot \{v_2\}^* \cdots \{v_m\}^*$ is written on tape T_0.

In the third phase, M reads tape T_0. In its finite control, M stores the "current voter" whose bribery money is read. For each voter v_i, a counter c_i is provided (within the finite memory of M). If a symbol v_i is read, c_i is incremented, and then M simultaneously moves all heads on the tapes T_j, where j is contained in $W_i(c_i)$. Hence, the string on tape T_0 is being processed in at most B/n (simultaneous) steps.

Finally, it is checked if the left border is reached (again) for all tapes T_j, $j > 0$. This is the case if and only if the guessed bribery was successful. □

The $W[2]$-hardness proof for VB-AM-PLP is analogous.

Recall that VB-SM-PLP is the same as VB-AM-PLP if the discretization level is zero. So, we conclude:

Corollary 5.4. VB-AM-PLP *parameterized by the budget per issue is* $W[2]$*-hard.*

Membership in $W[2]$ is an open problem for VB-AM-PLP when parameterized by the budget per issue. In contrast, we show definitive parameterized complexity results for different parameterizations. Refining and re-analyzing the proof of Theorem 5.3, we can prove:

Theorem 5.5. VB-SM-PLP *and* VB-AM-PLP *parameterized by the budget per issue and by the discretization level are* $W[2]$*-complete.*

5.2. Probabilistic lobbying with issue weighting

Recall from Theorem 4.4 that MB-SM-PLP-WIW and MB-AM-PLP-WIW are NP-complete. We now show that each of these problems is fixed-parameter tractable when parameterized by the budget. To this end, recall the Knapsack problem that was defined in the proof of Theorem 4.4: Given two finite lists of binary encoded integers, $(c_i)_{i=1}^n$ (a list of costs) and $(p_i)_{i=1}^n$ (a list of profits) associated to a list $(o_i)_{i=1}^n$ of objects, as well as two further integers, C and P (both encoded in binary), the question is whether there is a subset J of $\{1, \ldots, n\}$ such that $\sum_{i \in J} c_i \leq C$ and $\sum_{i \in J} p_i \geq P$. Thus, putting all objects from $\{o_j \mid j \in J\}$ into your knapsack does not violate your cost constraint C but does satisfy your profit demand P. Knapsack is an NP-hard problem that allows a pseudo-polynomial time algorithm. More precisely, this means that if all cost or all profit values are given in unary, a polynomial-time algorithm can be provided by using dynamic programming (see [57] for details). This yields FPTAS results both for the minimization version Min-Knapsack (where the goal is to minimize the costs, subject to the profit lower bound) and for the maximization version Max-Knapsack (where the goal is to maximize the profits, subject to the cost upper bound).

Theorem 5.6. MB-SM-PLP-WIW *and* MB-AM-PLP-WIW *parameterized by the budget or by the objective are in* FPT.

Proof. Since the unweighted variants of both problems are in P, we can compute the amount d_j of dollars to be spent to win referendum r_j in polynomial time in both cases. Namely, d_j is the smallest budget that is necessary to win in the instance that is derived from the given one by deleting everything that is independent of referendum r_j. Hence, the interesting cases are the weighted ones. We re-interpret the given MB-Y-PLP-WIW instance, where $Y \in \{\text{SM}, \text{AM}\}$, as a Knapsack instance.

In the MB-Y-PLP-WIW instance, every issue r_j has an associated cost d_j and weight w_j. The aim is to find a set of issues, i.e., a set $J \subseteq \{1, \ldots, n\}$, such that $\sum_{j \in J} d_j \leq B$ and $\sum_{j \in J} w_j \geq O$. Consider r_j as an object o_j in a Knapsack instance with cost $c_j = d_j$ and profit $p_j = w_j$, with the bounds $C = B$ and $P = O$. Then the $J \subseteq \{1, \ldots, n\}$ that is a solution to the MB-Y-PLP-WIW instance is also a solution to the Knapsack instance, and vice-versa. Furthermore, the pseudo-polynomial algorithm that solves Knapsack in time $\mathcal{O}(n2^{|B|})$, where $|B|$ denotes the length of the encoding of B, also solves MB-Y-PLP-WIW. Alternatively, we can use the well-known pseudo-polynomial algorithm to solve Knapsack in time $\mathcal{O}(n2^{|O|})$, where $|O|$ denotes the length of the encoding of O. □

Voter bribery with issue weighting keeps its complexity status for both evaluation criteria.

Since we can incorporate issue weights into brute-force computations, we have the following corollary to Theorems 5.1 and 5.2.

Corollary 5.7. 1. VB-SM-PLP-WIW *and* VB-AM-PLP-WIW *parameterized by the budget and by the discretization level are in* FPT.
2. VB-SM-PLP-WIW *parameterized by the budget is in* FPT.

It is not hard to transfer the $W[2]$-hardness results from the unweighted to the weighted case. However, it is unclear to us if or how the membership proofs of the preceding section transfer. The difficulty appears to lie in the weights that the reducing machine or the produced Turing machine would have to handle. Since it is not known in advance which items will be bribed to meet the objective requirement O, the summation of item weights cannot be performed by the reducing machine, but must be done by the produced nondeterministic multi-tape Turing machine. However, this Turing machine may only use time that can be measured in the parameter, which has been budget per issue in the unweighted case. We do not see how to do this. Therefore, we can state only the following.

Corollary 5.8. VB-SM-PLP-WIW *and* VB-AM-PLP-WIW *parameterized by the budget per issue are* $W[2]$*-hard.*

The proof of the following theorem is similar to the one of Theorem 5.3, although being a bit more technical. Details can be found in the Appendix.

Theorem 5.9. VB-SM-PLP-WIW *parameterized by the budget per issue and by the objective is in* $W[2]$.

It might be that $W[2]$ is not the smallest class in the W-hierarchy where the problem discussed in the preceding theorem could be placed. However, we do not know how to find an FPT or $W[1]$ algorithm for it, even in the case when all weights equal one. This is in contrast to the possibly related problem Partial t-Domination, which asks whether there is a set of at most k vertices in a graph that dominates at least t vertices. Our belief that these two problems are related is motivated by the fact that the classical dominating set problem was the starting point of the reduction showing hardness for Optimal Lobbying. Kneis, Mölle, and Rossmanith showed that the problem Partial t-Domination is in FPT even when parameterized by the threshold parameter t alone [58].

6. Approximability

As seen in Tables 1 and 2, many problem variants of probabilistic lobbying are NP-complete. Hence, it is interesting to study them not only from the viewpoint of parameterized complexity, but also from the viewpoint of approximability.

The budget constraint on the bribery problems studied so far gives rise to natural minimization problems: Try to minimize the amount spent on bribing. For clarity, let us denote these minimization problems by prefixing the problem name with Min, leading to, e.g., Min-OL.

6.1. Voter bribery is hard to approximate

The already-mentioned reduction of Christian et al. [2] (that proved that OL is $W[2]$-hard) is parameter-preserving (regarding the budget). The reduction also has the property that a possible solution found in the OL instance can be re-interpreted as a solution to the Dominating Set instance that the reduction started with, and the OL solution and the Dominating Set solution are of the same size. This in particular means that inapproximability results for Dominating Set transfer to inapproximability results for OL. Similar observations are true for the interrelation of Set Cover and Dominating Set, as well as for OL and VB-SM-PLP-WIW (or VB-AM-PLP-WIW).

The known inapproximability results [59,60] (see also [61] for a survey) for Set Cover hence give the following result (see also Footnote 4 in [3]).

Theorem 6.1. *There is a constant* $c > 0$ *such that* Min-OL *is not approximable within factor* $c \cdot \log n$ *(where* n *denotes the number of issues) unless* NP $\subset$ DTIME$(n^{\log\log n})$.

Since OL can be viewed as a special case of both problem VB-Y-PLP and of problem VB-Y-PLP-WIW for $Y \in \{\text{SM}, \text{AM}\}$, we have the following corollary.

Corollary 6.2. *For* $Y \in \{\text{SM}, \text{AM}\}$*, there is a constant* $c_Y > 0$ *such that both* Min-VB-Y-PLP *and* Min-VB-Y-PLP-WIW *are not approximable within factor* $c_Y \cdot \log n$ *unless* NP *is contained in* DTIME$(n^{\log\log n})$*, where* n *denotes the number of issues.*

Proof. The proof of Theorem 4.3 explains in detail how to interpret an instance of OL as a VB-Y-PLP instance, $Y \in \{\text{SM}, \text{AM}\}$. The relation $B = n \cdot b$ between the budget B and the number of voters b holds for both optimum and approximate solutions. Hence, the n is canceled out when looking at the approximation ratio. □

Conversely, a logarithmic-factor approximation can be given for the minimum-budget versions of all our problems, as we will show now. We first discuss the relation to the well-known Set Cover problem, sketching a tempting, yet flawed reduction and pointing out its pitfalls. Avoiding these pitfalls, we then give an approximation algorithm for Min-VB-AM-PLP. Moreover, we define the notion of cover number, which allows us to state inapproximability results for Min-VB-AM-PLP. Similar results hold for Min-VB-SM-PLP, the constructions are sketched at the end of the section.

Voter bribery problems are closely related to set cover problems,[9] in particular in the average-majority scenario, so that we should be able to carry over approximability ideas from that area. The intuitive translation of a Min-VB-AM-PLP instance into a Set Cover instance is as follows: The universe of the derived Set Cover instance should be the set of issues, and the sets (in the Set Cover instance) are formed by considering the sets of issues that could be influenced (by changing a voter's opinion) through bribery of a specific voter. Namely, when we pay voter v a specific amount of money, say d dollars, he or she will credit d/n dollars to each issue and possibly change v's opinion (or at least raise v's acceptance probability to some "higher level"). The weights associated with the sets of issues correspond to the bribery costs that are (minimally) incurred to lift the issues in the set to some "higher level". There are four differences to classical set covering problems:

1. Multiple voters may cover the same set of issues (with different bribing costs).
2. The sets associated with one voter are not independent. For each voter, the sets of issues that can be influenced by bribing that voter are linearly ordered by set inclusion. Moreover, when bribing a specific voter, we have to first influence the "smaller sets" (which might be expensive) before possibly influencing the "larger ones"; so, weights are attached to set differences, rather than to sets.
3. A *cover number* $c(r_j)$ is associated with each issue r_j, indicating by how many levels voters must raise their acceptance probabilities in order to arrive at average majority for r_j. The cover numbers can be computed beforehand for a given instance. Then, we can also associate cover numbers with sets of issues (by summation), which finally leads to the cover number $N = \sum_{j=1}^{n} c(r_j)$ of the whole instance.
4. The money paid "per issue" might not have been sufficient for influencing a certain issue up to a certain level, but it is not "lost"; rather, it would make the next bribery step cheaper, hence (again) changing weights in the set cover interpretation.

To understand these connections better, let us have another look at our running example (under voter bribery with average-majority evaluation, i.e., Min-VB-AM-PLP), assuming an all-ones target vector. If we paid 30 dollars to voter v_1, he or she would credit 10 dollars to each issue, which would raise his or her acceptance probability for the second issue from 0.3 to 0.4; no other issue level is changed. Hence, this would correspond to a set containing only r_2 with weight 30. Note that by this bribery, the costs for raising the acceptance probability of voter v_1 to the next level would be lowered for the other two issues. For example, spending 15 more dollars on v_1 would raise r_3 from 0.5 to 0.6, since all in all 45 dollars have been spent on voter v_1, which means 15 dollars per issue. If the threshold is 60% in that example, then the first issue is already accepted (as desired by The Lobby), but the second issue has gone up from 0.5 to only 0.55 on average, which means that we have to raise either the acceptance probability of one voter by two levels (for example, by paying 210 dollars to voter v_1), or we have to raise the acceptance probability of each voter by one level (by paying 30 dollars to voter v_1 and another 30 dollars to voter v_2). This can be expressed by saying that the first issue has a cover number of zero, and the second has a cover number of two.

When we interpret an OL instance as a VB-AM-PLP instance, the cover number of the resulting instance equals the number of issues, assuming that the votes for all issues need amendment. Thus we have the following corollary:

Corollary 6.3. *There is a constant $c > 0$ such that* Min-VB- AM-PLP *is not approximable within factor $c \cdot \log N$ unless* NP $\subset$ DTIME($N^{\log\log N}$), *where N is the cover number of the given instance. A fortiori, the same statement holds for the problem* Min-VB-AM-PLP-WIW.

Let H denote the harmonic sum function, i.e., $H(r) = \sum_{i=1}^{r} 1/i$. It is well known that $H(r) = O(\log r)$. More precisely, it is known that

$$\ln r \leq H(r) \leq \ln r + 1.$$

We now show the following theorem.

Theorem 6.4. Min-VB-AM-PLP *can be approximated within a factor of* $\ln(N) + 1$, *where N is the cover number of the given instance.*

Proof. Consider the greedy algorithm shown in Fig. 1, where t is the given threshold and we assume that The Lobby has the all-ones target vector. Note that the cover numbers (per referendum) can be computed from the cost matrix C_P and the threshold t before calling the algorithm the very first time.

Observe that our greedy algorithm influences voters by raising their acceptance probabilities by only one level, so that the amount d_v possibly spent on voter v in Step 3 of the algorithm actually corresponds to a set of referenda; we do not have to consider multiplicities of issues (raised over several levels) here.

[9] As pointed out by a reviewer, our lobbying problems are also closely related to the problem of approximating winners in the Dodgson voting systems, as they too can be seen as extensions of the Set Cover problem. Indeed, our approximation algorithm in Fig. 1 is similar to that due to Caragiannis et al. [62].

Input: A probability matrix $P \in \mathbb{Q}_{[0,1]}^{m\times n}$ (implicitly specifying a set V of m voters and a set R of n referenda), a cost matrix C_P, a treshold t, and n cover numbers $c(r_1), \ldots, c(r_n) \in \mathbb{N}$.

1. Delete referenda that are already won (indicated by $c(r_j) = 0$), and modify R and C_P accordingly.
2. If $R = \emptyset$ then output the amount spent on bribing so far and STOP.
3. For each voter v, compute the least amount of money, d_v, that could raise any level in C_P. Let n_v be the number of referenda whose levels are raised when spending d_v dollars on voter v.
4. Bribe voter v such that d_v/n_v is minimum.
5. Modify C_P by subtracting d_v/n_v from each amount listed for voter v.
6. Modify c by subtracting one from $c(r)$ for those referenda $r \in R$ influenced by this bribery action.
7. Recurse.

Fig. 1. Greedy approximation algorithm for MIN-VB-AM-PLP in Theorem 6.4.

Let ℓ be the number of times Step 3 of the greedy bribery algorithm is executed. Let $S_1, \ldots, S_\ell$ be the sequence of sets of referenda influenced by the greedy bribery algorithm, along with the sequence $v_1, \ldots, v_\ell$ of voters and the sequence $d_1, \ldots, d_\ell$ of bribery dollars spent this way. Let $R_1 = R, \ldots, R_\ell, R_{\ell+1} = \emptyset$ be the corresponding decreasing sequence of current sets of referenda yet to be won, with the accordingly modified cover numbers c_i. Let $j(r, k)$ denote the index of the set in the sequence influencing referendum r the kth time with $k \le c(r)$, i.e., $r \in S_{j(r,k)}$ and $|\{i < j(r, k) \mid r \in S_i\}| = k - 1$. To cover r the kth time, we have to pay $\chi(r, k) = d_{j(r,k)}/|S_{j(r,k)}|$ dollars. The greedy algorithm will incur a cost of $\chi_{greedy} = \sum_{r\in R} \sum_{k=1}^{c(r)} \chi(r, k)$ in total.

An alternative view of the greedy algorithm is from the perspective of the referenda: By running the algorithm, we implicitly define a sequence $s_1, \ldots, s_N$ of referenda, where $N = c(R) = \sum_{r\in R} c(r)$ is the cover number of the original instance, such that $S_1 = \{s_{\lambda(1)}, \ldots, s_{\rho(1)}\}, S_2 = \{s_{\lambda(2)}, \ldots, s_{\rho(2)}\}, \ldots, S_\ell = \{s_{\lambda(\ell)}, \ldots, s_{\rho(\ell)}\}$, where $\lambda, \rho : \{1, \ldots, \ell\} \to \{1, \ldots, N\}$ are functions such that $\lambda(i)$ gives the element of S_i with the smallest subscript and $\rho(i)$ gives the element of S_i with the greatest subscript for each i, $1 \le i \le \ell$:

$$\lambda(i) = 1 + \sum_{j<i} |S_j| \quad \text{and} \quad \rho(i) = \sum_{j\le i} |S_j|.$$

Ties (how to list elements within any S_i) are broken arbitrarily.

Consider s_j with $\lambda(i) \le j \le \rho(i)$. Slightly abusing notation, we associate a cost $\chi'(s_j) = d_i/|S_i|$ with S_i for each i (keeping in mind the multiplicities of covering implied by the sequence $\langle S_i\rangle_i$), so that $d_i = \sum_{\lambda(i)\le j\le\rho(i)} \chi'(s_j)$ and hence

$$\chi_{greedy} = \sum_{k=1}^{\ell} d_k = \sum_{j=1}^{N} \chi'(s_j). \tag{1}$$

The current referendum set R_i has cover number

$$c(R_i) = c_i(R_i) = N - \lambda(i) + 1 \le N - j + 1, \tag{2}$$

as $\lambda(i) \le j$.

Let χ_{opt} be the cost of an optimum bribery strategy $\mathscr{B}^*$ of the original universe. $\mathscr{B}^*$ also yields a cover of any referendum set R_i, $1 \le i \le \ell$, with cost at most χ_{opt}. The average cost per element (taking into account multiplicities as given by the cover numbers) is at most $\chi_{opt}/c(R_i)$. (So, whether or not some new levels are obtained through bribery does not really matter here.)

The strategy $\mathscr{B}^*$ can also be described by a sequence of sets of referenda $S_1^*, \ldots, S_q^*$, with corresponding voters $v_1^*, \ldots, v_q^*$ and dollars $d_1^*, \ldots, d_q^*$ spent. Hence, $\chi_{opt} = \sum_{\kappa=1}^{q} d_\kappa^*$. With each bribery step we associate the cost factor $\chi'_{opt}(r) = d_\kappa^*/|S_\kappa^*|$ for each issue r contained in S_κ^*.

Recall that $\mathscr{B}^*$ could be also viewed as a bribery strategy for R_i. By the pigeon hole principle, there is a referendum r in R_i (to be influenced the kth time in $\mathscr{B}^*$) with cost factor of

$$\chi'_{opt}(r) = \frac{d^*_\kappa}{|S^*_\kappa|} \leq \frac{d^*_\kappa}{|S^*_\kappa \cap R_i|} \leq \frac{\chi_{opt}}{c(R_i)}, \tag{3}$$

where κ is the index such that S^*_κ contains r for the kth time in $\mathscr{B}^*$ (usually, the cost would be smaller, since part of the bribery has already been paid before).

Since (S_i, v_i) was picked by our greedy algorithm in the situation described by R_i so as to minimize $d_i/|S_i|$, we find with Eq. (3) for any s_j with $\lambda(i) \leq j \leq \rho(i)$:

$$\chi'(s_j) = \frac{d_i}{|S_i|} \leq \frac{d^*_\kappa}{|S^*_\kappa \cap R_i|} \leq \frac{\chi_{opt}}{c(R_i)}. \tag{4}$$

We conclude from Eq. (4) together with Eq. (2) that

$$\chi'(s_j) \leq \frac{\chi_{opt}}{c(R_i)} = \frac{\chi_{opt}}{N - \lambda(i) + 1} \leq \frac{\chi_{opt}}{N - j + 1}.$$

Hence, due to Eq. (1),

$$\chi_{greedy} = \sum_{j=1}^{N} \chi'(s_j) \leq \sum_{j=1}^{N} \frac{\chi_{opt}}{N - j + 1} = H(N)\chi_{opt} \leq (\ln(N) + 1)\chi_{opt},$$

which completes the proof. □

In the strict-majority scenario (SM), cover numbers would have a different meaning—we thus call them *strict cover numbers*: For each referendum, the corresponding strict cover number tells in advance how many voters have to change their opinions (bringing them individually over the given threshold t) to accept this referendum. Again, the strict cover number of a problem instance is the sum of the strict cover numbers of all given referenda.

The corresponding greedy algorithm would therefore choose to influence voter v_i (with d_i dollars) in the ith loop so that v_i changes his or her opinion on some referendum r_j such that $d_i/|\rho_j|$ is minimized.[10]

We can now read the approximation bound proof given for the average-majority scenario nearly literally as before, by re-interpreting the formulation "influencing referendum r" meaning now a complete change of opinion for a certain voter (not just gaining one level somehow). This establishes the following result.

Theorem 6.5. Min-VB-SM-PLP *can be approximated within a factor of* $\ln(N) + 1$, *where N is the strict cover number of the given instance.*

Note that this result is in some sense stronger than Theorem 6.4 (which refers to the average-majority scenario), since the cover number of an instance could be larger than the strict cover number.

This approximation result is complemented by a corresponding hardness result.

Theorem 6.6. *There is a constant $c > 0$ such that* Min-VB-SM-PLP *is not approximable within factor $c \cdot \log N$ unless* NP $\subset$ DTIME($N^{\log\log N}$), *where N is the strict cover number of the given instance. A fortiori, the same statement holds for* Min-VB-SM-PLP-WIW.

Unfortunately, those greedy algorithms do not (immediately) transfer to the case when issue weights are allowed. These weights might also influence the quality of approximation, but a simplistic greedy algorithm might result in covering the "wrong" issues. Also, the proof of the approximation factor given above will not carry over, since we need as one of the proof's basic ingredients that an optimum solution can be interpreted as a partial one at some point. Those problems tend to have a different flavor.

6.2. Polynomial-time approximation schemes

Those problems for which we obtained FPT results in the case of issue weights actually enjoy a fully polynomial-time approximation scheme (FPTAS) when viewed as minimization problems.[11] The proof of Theorem 5.6 yields an FPTAS. That result was obtained by transferring pseudo-polynomial time algorithms: For each fixed value of the parameter (the budget B or the objective O), we obtain a polynomial-time algorithm for the decision problem, which can be used to approximate the minimization problem.

[10] Possibly, there is a whole set ρ_j of referenda influenced this way.

[11] A fully polynomial-time approximation scheme is an algorithm that for each pair (x, ε), where x is an instance of an optimization problem and $\varepsilon > 0$ is a rational constant, runs in time polynomial in $|x|$ and in $1/\varepsilon$ and outputs an approximate solution for x within a factor of $1 + \varepsilon$.

Theorem 6.7. *Both* Min-MB-SM-PLP-WIW *and* Min-MB-AM-PLP-WIW *admit an FPTAS.*

Proof. We provide some of the details for Min-MB-SM-PLP-WIW only. As in the proof of Theorem 5.6, we first compute the amount, d_j, to be spent to win referendum r_j in polynomial time. We then re-interpret the given instance of Min-MB-SM-PLP-WIW as a Min-Knapsack instance. Then the $J \subseteq \{1, \ldots, n\}$ that is a solution to the MB-SM-PLP-WIW instance is also a solution to the Knapsack instance, and vice-versa. Furthermore, the FPTAS algorithm that approximates Min-Knapsack also gives an FPTAS for MB-SM-PLP-WIW. □

Let us mention here that the issue-weighted problem variants are actually bicriteria problems: We want to achieve as much as possible (expressed by the objective O) and pay as little as possible (expressed by the budget B). So, we could also consider this as a maximization problem (where now B becomes again part of the input). By the close relation to Knapsack mentioned above, the pseudo-polynomial time algorithms again result in PTAS results for this model:

Theorem 6.8. *Both* Max-MB-SM-PLP-WIW *and* Max-MB-AM-PLP-WIW *admit an FPTAS.*

It would be interesting to study this optimization criterion in the light of other bribery scenarios.

7. Conclusions

This paper lies at the intersection of three research streams: computational social choice, reasoning under uncertainty, and computational complexity. It extends the study of lobbying initiated by Christian et al. [2] to probabilistic settings. We have shown that uncertainty complicates the picture; the choice of model strongly affects computational complexity.

We have studied four lobbying scenarios in a probabilistic setting, both with and without issue weights. Among these, we identified problems that can be solved in polynomial time, problems that are NP-complete yet fixed-parameter tractable, and problems that are hard (namely, $W[2]$-complete or $W[2]$-hard) in terms of their parameterized complexity with suitable parameters. We also investigated the approximability of hard probabilistic lobbying problems (without issue weights) and obtained both approximation and inapproximability results.

Our research in computational complexity of the mentioned problems falls into three categories: (1) classical complexity (i.e., mainly NP-completeness results), (2) approximability and (3) parameterized complexity.

An interesting direction for future work would be to study the parameterized complexity of such problems under different parameterizations. We would also like to investigate the open question of whether one can find logarithmic-factor approximations for voter bribery with issue weights. It would be also interesting to design parameterized approximation algorithms (a new trend bridging the areas of parameterized and of approximation algorithms) in particular for those problems where we cannot expect FPT results and (at the same time) APX results. We refer the interested reader to the survey paper by Marx [63].

From the viewpoint of parameterized complexity, it would be interesting to solve the questions left open in this paper (see the question marks in Tables 1 and 2), in particular regarding voter bribery under average majority, with and without issue weighting, parameterized by the budget or by the budget per issue. The parameterized complexity of the problems involving microbribery with issue weighting, under either strict majority or average majority, parameterized by the budget per issue, is an open issue as well (again, see Table 2). In fact, we did not see how to put these problems in any level of the W-hierarchy. Problems that involve numbers seem to have a particular flavor that makes them hard to tackle with these techniques [64], but we suggest this as the subject of further studies. Note that all probabilistic lobbying problems that have been connected with $W[2]$ somehow in this paper are in fact strongly NP-hard, i.e., their hardness does not depend on whether the numbers in their inputs are encoded in unary or in binary. We suspect that this adds to the difficulty when dealing with these problems from a parameterized perspective.

From the viewpoint of Social Choice, it would be interesting to introduce and discuss other scenarios that involve randomness in some way. Our approach is only one among many different possible venues, inspired by the model underlying Optimal Lobbying.

Acknowledgments

The second and sixth authors were supported in part by DFG grants RO 1202/11-1, RO 1202/12-1 (within the European Science Foundation's EUROCORES program LogICCC), and RO 1202/15-1, and the Alexander von Humboldt Foundation's TransCoop program. The fourth and fifth authors were supported in part by NSF grants CCF-1049360 and ITR-0325063. The second author was supported in part by National Research Foundation (Singapore) under grant NRF-RF 2009-08 and DFG grant ER 738/1-1.

Appendix. Two proofs of membership in $W[2]$

We collect here two proofs that are similar to the one of Theorem 5.3 and that might be of less interest, although several details have to be taken care of.

A.1. A detailed proof of Theorem 5.5

Proof. As mentioned above, we already have hardness for a discretization level of zero, i.e., when the second parameter is fixed to the lowest possible value. We implicitly show membership in $W[2]$ for VB-SM-PLP in Theorem 5.3 when the second parameter only plays a role in the polynomial part of the run time estimate. Hence, the claim is a simple corollary of what we have already shown above.

It remains to prove membership in $W[2]$ for VB- AM-PLP. This can be seen by modifying the proof of Theorem 5.3. To show membership in $W[2]$, we reduce VB-AM-PLP to Short Multi-tape Nondeterministic Turing Machine Computation. So, it suffices to describe how a nondeterministic multi-tape Turing machine can solve such a lobbying problem.

Consider an instance of VB-SM-PLP: a probability matrix $P \in \mathbb{Q}_{[0,1]}^{m \times n}$ with a cost matrix C_P, a budget B, and a fixed threshold t. Again, we may identify t with a certain step level for the price functions.

From this input, the reducing machine computes the following:

- It does some preprocessing, so that it is guaranteed that the overall money that could be meaningfully spent on any voter is bounded by $\lceil B/n \rceil$, which is the first parameter. No larger amount of money is available, anyhow. Confidence steps that cannot be reached at all are modeled by requiring an investment of (all in all) $\lfloor B/n \rfloor + 1$ dollars on that voter for each issue.
- It computes the entity $s(j)$ that now denotes the number of confidence steps issue r_j has to be raised in total to ensure a win of that referendum. Notice that $s(j)$ can be assumed to be bounded by the product of the first parameter, more precisely, by $\lfloor B/n \rfloor + 1$ (as argued before), and the second parameter, more precisely, by $k + 1$: Each bribe (to whatever voter v) of n dollars (recall that we can again rely on bribery money being used in multiples of n, the number of issues) will raise v's confidence in voting according to the agenda of The Lobby by at most $k + 1$ steps.
- It finally computes $W_i(\ell)$ which now gives the list of issues whose confidence is raised when investing $\ell \cdot n$ dollars on v_i (compared to $(\ell - 1) \cdot n$ dollars), plus the number of confidence steps by which the corresponding issue is raised.

The nondeterministic multi-tape Turing machine M we describe next has, in particular, access to $W_i(\ell)$ and to $s(j)$. M has $n + 1$ working tapes T_j, $0 \leq j \leq n$, all except one of which correspond to issues r_j, $1 \leq j \leq n$. We will use the set of voters, $V = \{v_1, \ldots, v_m\}$, as part of the work alphabet. The (formal) input tape of M is ignored.

M starts by writing $s(j)$ symbols # onto tape j for each j, $1 \leq j \leq n$. By using simultaneous writing steps, this needs at most $f(B/n, k) := (\lfloor B/n \rfloor + 1)\,(k + 1)$ steps, since $s(j) \leq f(B/n, k)$ as argued above. The "information hiding" trick works as before.

Second, for each $i \in \{1, \ldots, m\}$, M writes k_i symbols v_i from the alphabet V on the zeroth tape, T_0, such that $\sum_{i=1}^{m} k_i \leq B/n$. This is the nondeterministic guessing phase where the amount of bribery money spent on each voter, namely $k_i \cdot n$ for voter v_i, is determined. The finite control is used to ensure that a word from the language $\{v_1\}^* \cdot \{v_2\}^* \cdots \{v_m\}^*$ is written on tape T_0.

In the third phase, M reads tape T_0. In its finite control, M stores the "current voter" whose bribery money is read. For each voter v_i, a counter c_i is provided (within the finite memory of M). If a symbol v_i is read, c_i is incremented, and then M simultaneously moves all heads on the tapes T_j, where j is contained in $W_i(c_i)$; notice that the number of steps each head has to move is now also stored in $W_i(c_i)$. Hence, the string on tape T_0 is being processed in at most $f\,(B/n, k)$ (simultaneous) steps.

Finally, it is checked if the left border is reached (again) for all tapes T_j, $j > 0$. This is the case if and only if the guessed bribery was successful. □

A.2. Proof of Theorem 5.9

Proof. Membership in $W[2]$ is a bit more tricky than in the unweighted case from Theorem 5.3. To show membership in $W[2]$, we reduce VB-SM-PLP-WIW to Short Multi-tape Nondeterministic Turing Machine Computation. We describe how a nondeterministic multi-tape Turing machine M can solve such a lobbying problem.

Consider an instance of VB-SM-PLP-WIW: a probability matrix $P \in \mathbb{Q}_{[0,1]}^{m \times n}$ with a cost matrix C_P, a budget B, and a fixed threshold t, as well as an objective O. We may again identify t with a certain step level for the price functions.

We describe the work of the reducing machine in the following.

- The reducing machine calculates the difference O' between the target weight and the sum of the weights of the referenda that are already won.
- The reducing machine replaces issue weights bigger than O' with $O' + 1$.
- The reducing machine eliminates all referenda that are already won.
- For each referendum that is not already won, the reducing machine introduces a special letter r_i to be used on tape T_0 of the Turing machine to be constructed.

For simplicity, we assume in the following that $O' = O$, i.e., the reduction steps described above have not changed the instance.

After these steps, all the weights are bounded by the second parameter. Moreover, recall that by the problem definition, the weight associated to each issue is at least one.

Otherwise, the reducing machine works as described in the proof of Theorem 5.3. In particular, from P, C_P, and t, the machine extracts the information $H_{i,j}(d)$, $1 \leq d \leq B$, and then $s(j)$ (per r_j) and finally the *winning table* $W_i(\ell)$.

We now describe how the Turing machine M works.

- M begins by guessing $\nu \leq O$ referenda that should also be won. (Recall that the weight associated to each issue is at least one, so guessing more than O referenda is not necessary.) Then, M spends $\mathcal{O}(f(O))$ time calculating whether winning those guessed referenda $r_{\gamma(1)}, \ldots, r_{\gamma(\nu)}$, $\nu \leq O$, would be sufficient to exceed the threshold O. If not, M halts and rejects, otherwise it continues working.
- The mentioned threshold test can be implemented as follows: M guesses at most O referenda and writes the corresponding symbols on tape T_0. It first checks if all guessed referenda are pairwise different. If not, M stops and rejects. Otherwise, M writes a special symbol $O-1$ times on the second tape, T_1, and moves its head on this tape to the right. Then, it moves its head w_j steps to the left on tape T_1 (replacing the special symbol by the blank symbol again) upon reading symbol r_j on tape T_0. The threshold O is passed if and only if the blank symbol is read on T_1 after processing $r_{\gamma(\nu)}$ on tape T_0. Notice that tape T_1 will be empty again after this phase, while tape T_0 will still contain the guessed referenda. Observe that the time needed by the Turing machine M depends only on O.
- M will then continue to work as described in the proof of Theorem 5.3. During this phase, the weights will be completely ignored. In particular, this part of M can be produced in polynomial time by the reducing machine. As can be seen in the proof of Theorem 5.3, the time needed by M in this phase depends only on B/n.
- Finally, M will verify if all (at most O) referenda guessed initially have been won. In contrast to the proof of Theorem 5.3, this test can now be implemented in a sequential fashion (upon reading r_j on tape T_0, the corresponding test is performed on tape T_j), needing time dependent on O.

This concludes the proof. □

References

[1] G. Erdélyi, H. Fernau, J. Goldsmith, N. Mattei, D. Raible, J. Rothe, The complexity of probabilistic lobbying, in: Proceedings of the 1st International Conference on Algorithmic Decision Theory, in: Lecture Notes in Artificial Intelligence #5783, Springer-Verlag, 2009, pp. 86–97.
[2] R. Christian, M. Fellows, F. Rosamond, A. Slinko, On complexity of lobbying in multiple referenda, Review of Economic Design 11 (3) (2007) 217–224.
[3] T. Sandholm, S. Suri, A. Gilpin, D. Levine, Winner determination in combinatorial auction generalizations, in: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems, ACM Press, 2002, pp. 69–76.
[4] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Llull and Copeland voting computationally resist bribery and constructive control, Journal of Artificial Intelligence Research 35 (2009) 275–341.
[5] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, How hard is bribery in elections?, Journal of Artificial Intelligence Research 35 (2009) 485–532.
[6] J. Reinganum, A formal theory of lobbying behaviour, Optimal Control Applications and Methods 4 (4) (1983) 71–84.
[7] J. von Neumann, O. Morgenstern, Theory of Games and Economic Behavior, Princeton University Press, 1944.
[8] M. Baye, D. Kovenock, C. de Vries, Rigging the lobbying process: an application of the all-pay auction, The American Economic Review 83 (1) (1993) 289–294.
[9] M. Baye, D. Kovenock, C. de Vries, The all-pay auction with complete information, Economic Theory 8 (2) (1996) 291–305.
[10] C. Crombez, Information, lobbying and the legislative process in the European union, European Union Politics 3 (1) (2002) 7–32.
[11] E. Ephrati, J. Rosenschein, A heuristic technique for multi-agent planning, Annals of Mathematics and Artificial Intelligence 20 (1–4) (1997) 13–67.
[12] S. Ghosh, M. Mundhe, K. Hernandez, S. Sen, Voting for movies: the anatomy of recommender systems, in: Proceedings of the 3rd Annual Conference on Autonomous Agents, ACM Press, 1999, pp. 434–435.
[13] C. Dwork, R. Kumar, M. Naor, D. Sivakumar, Rank aggregation methods for the web, in: Proceedings of the 10th International World Wide Web Conference, ACM Press, 2001, pp. 613–622.
[14] J. Bartholdi III, C. Tovey, M. Trick, Voting schemes for which it can be difficult to tell who won the election, Social Choice and Welfare 6 (2) (1989) 157–165.
[15] E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP, Journal of the ACM 44 (6) (1997) 806–825.
[16] J. Rothe, H. Spakowski, J. Vogel, Exact complexity of the winner problem for Young elections, Theory of Computing Systems 36 (4) (2003) 375–386.
[17] E. Hemaspaandra, H. Spakowski, J. Vogel, The complexity of Kemeny elections, Theoretical Computer Science 349 (3) (2005) 382–391.
[18] M. Brill, F. Fischer, The price of neutrality for the ranked pairs method, in: Proceedings of the 26th AAAI Conference on Artificial Intelligence, AAAI Press, 2012, pp. 1299–1305.
[19] J. Bartholdi III, C. Tovey, M. Trick, The computational difficulty of manipulating an election, Social Choice and Welfare 6 (3) (1989) 227–241.
[20] J. Bartholdi III, J. Orlin, Single transferable vote resists strategic voting, Social Choice and Welfare 8 (4) (1991) 341–354.
[21] V. Conitzer, T. Sandholm, Universal voting protocol tweaks to make manipulation hard, in: Proceedings of the 18th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, 2003, pp. 781–788.
[22] E. Elkind, H. Lipmaa, Small coalitions cannot manipulate voting, in: Proceedings of the 9th International Conference on Financial Cryptography and Data Security, in: Lecture Notes in Computer Science #3570, Springer-Verlag, 2005, pp. 285–297.
[23] V. Conitzer, T. Sandholm, J. Lang, When are elections with few candidates hard to manipulate?, Journal of the ACM 54 (3) (2007) Article 14.
[24] E. Hemaspaandra, L. Hemaspaandra, Dichotomy for voting systems, Journal of Computer and System Sciences 73 (1) (2007) 73–83.
[25] A. Procaccia, J. Rosenschein, Junta distributions and the average-case complexity of manipulating elections, Journal of Artificial Intelligence Research 28 (2007) 157–181.
[26] E. Brelsford, P. Faliszewski, E. Hemaspaandra, H. Schnoor, I. Schnoor, Approximability of manipulating elections, in: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, AAAI Press, 2008, pp. 44–49.
[27] R. Meir, A. Procaccia, J. Rosenschein, A. Zohar, Complexity of strategic behavior in multi-winner elections, Journal of Artificial Intelligence Research 33 (2008) 149–178.
[28] M. Zuckerman, A.D. Procaccia, J.S. Rosenschein, Algorithms for the coalitional manipulation problem, Artificial Intelligence 173 (2) (2009) 392–412.
[29] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, The shield that never was: societies with single-peaked preferences are more open to manipulation and control, in: Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge, ACM Press, 2009, pp. 118–127.
[30] J. Davies, G. Katsirelos, N. Narodytska, T. Walsh, Complexity of and algorithms for Borda manipulation, in: Proceedings of the 25th AAAI Conference on Artificial Intelligence, AAAI Press, 2011, pp. 657–662.
[31] N. Narodytska, T. Walsh, L. Xia, Manipulation of Nanson's and Baldwin's rules, in: Proceedings of the 25th AAAI Conference on Artificial Intelligence, AAAI Press, 2011, pp. 713–718.

[32] N. Mattei, J. Forshee, J. Goldsmith, An empirical study of voting rules and manipulation with large datasets, in: F. Brandt, P. Faliszewski (Eds.), Proceedings of the 4th International Workshop on Computational Social Choice, AGH University of Science and Technology, Kraków, Poland, 2012, pp. 299–310.
[33] J. Bartholdi III, C. Tovey, M. Trick, How hard is it to control an election?, Mathematical and Computer Modelling 16 (8–9) (1992) 27–40.
[34] E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Anyone but him: the complexity of precluding an alternative, Artificial Intelligence 171 (5–6) (2007) 255–285.
[35] E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Hybrid elections broaden complexity-theoretic resistance to control, Mathematical Logic Quarterly 55 (4) (2009) 397–424.
[36] G. Erdélyi, M. Nowak, J. Rothe, Sincere-strategy preference-based approval voting fully resists constructive control and broadly resists destructive control, Mathematical Logic Quarterly 55 (4) (2009) 425–443.
[37] G. Erdélyi, J. Rothe, Control complexity in fallback voting, in: Proceedings of Computing: the 16th Australasian Theory Symposium, in: Conferences in Research and Practice in Information Technology Series, Australian Computer Society, 2010, pp. 39–48.
[38] G. Erdélyi, L. Piras, J. Rothe, The complexity of voter partition in Bucklin and fallback voting: solving three open problems, in: Proceedings of the 10th International Joint Conference on Autonomous Agents and Multiagent Systems, IFAAMAS, 2011, pp. 837–844.
[39] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, A richer understanding of the complexity of election systems, in: S. Ravi, S. Shukla (Eds.), Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz, Springer, 2009, pp. 375–406 (Chapter 14).
[40] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, Using complexity to protect elections, Communications of the ACM 53 (11) (2010) 74–82.
[41] P. Faliszewski, A. Procaccia, AI's war on manipulation: are we winning?, AI Magazine 31 (4) (2010) 53–64.
[42] V. Conitzer, Making decisions based on the preferences of multiple agents, Communications of the ACM 53 (3) (2010) 84–94.
[43] D. Baumeister, G. Erdélyi, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Computational aspects of approval voting, in: J. Laslier, R. Sanver (Eds.), Handbook on Approval Voting, Springer, 2010, pp. 199–251 (Chapter 10).
[44] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Llull and Copeland voting broadly resist bribery and control, in: Proceedings of the 22nd AAAI Conference on Artificial Intelligence, AAAI Press, 2007, pp. 724–730.
[45] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Copeland voting fully resists constructive control, in: Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management, in: Lecture Notes in Computer Science #5034, Springer-Verlag, 2008, pp. 165–176.
[46] N. Mattei, J. Goldsmith, A. Klapper, On the complexity of bribery and manipulation in tournaments with uncertain information, in: Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference, 2012, pp. 549–554.
[47] R. Downey, M. Fellows, Parameterized Complexity, Springer-Verlag, Berlin, Heidelberg, New York, 1999.
[48] J. Flum, M. Grohe, Parameterized Complexity Theory, in: EATCS Texts in Theoretical Computer Science, Springer-Verlag, 2006.
[49] R. Niedermeier, Invitation to Fixed-Parameter Algorithms, Oxford University Press, 2006.
[50] M. Cesati, The Turing way to parameterized complexity, Journal of Computer and System Sciences 67 (2003) 654–685.
[51] M. Fellows, Towards fully multivariate algorithmics: some new results and directions in parameter ecology, in: J. Fiala, J. Kratochvíl, M. Miller (Eds.), Proceedings of the 20th International Workshop on Combinatorial Algorithms, in: Lecture Notes in Computer Science #5874, Springer-Verlag, 2009, pp. 2–10.
[52] R. Niedermeier, Reflections on multivariate algorithmics and problem parameterization, in: J.-Y. Marion, T. Schwentick (Eds.), Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science, in: Leibniz International Proceedings in Informatics (LIPIcs), vol. 5, Schloss Dagstuhl—Leibniz-Zentrum für Informatik, 2010, pp. 17–32.
[53] L. Xu, F. Hutter, H. Hoos, K. Leyton-Brown, SATzilla: portfolio-based algorithm selection for SAT, Journal of Artificial Intelligence Research 32 (2008) 565–606.
[54] F. Hutter, H. Hoos, K. Leyton-Brown, T. Stützle, ParamILS: an automatic algorithm configuration framework, Journal of Artificial Intelligence Research 36 (2009) 267–306.
[55] C. Lindner, J. Rothe, Fixed-parameter tractability and parameterized complexity, applied to problems from computational social choice, in: A. Holder (Ed.), Mathematical Programming Glossary, INFORMS Computing Society, 2008.
[56] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, 1979.
[57] H. Kellerer, U. Pferschy, D. Pisinger, Knapsack Problems, Springer-Verlag, Berlin, Heidelberg, New York, 2004.
[58] J. Kneis, D. Mölle, P. Rossmanith, Partial vs. complete domination: t-dominating set, in: J. van Leeuwen, G. Italiano, W. van der Hoek, C. Meinel, H. Sack, F. Plasil (Eds.), Proceedings of the 33nd Conference on Current Trends in Theory and Practice of Computer Science, in: Lecture Notes in Computer Science #4362, Springer-Verlag, 2007, pp. 367–376.
[59] M. Bellare, S. Goldwasser, C. Lund, A. Russell, Efficient probabilistically checkable proofs and applications to approximations, in: Proceedings of the 25th ACM Symposium on Theory of Computing, ACM Press, 1993, pp. 294–304.
[60] R. Raz, S. Safra, A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP, in: Proceedings of the 29th ACM Symposium on Theory of Computing, ACM Press, 1997, pp. 475–484.
[61] S. Arora, C. Lund, Hardness of approximations, in: D. Hochbaum (Ed.), Approximation Algorithms for NP-Hard Problems, PWS Publishing Company, 1996, pp. 399–446 (Chapter 10).
[62] I. Caragiannis, J. Covey, M. Feldman, C. Homan, C. Kaklamanis, N. Karanikolas, A. Procaccia, J. Rosenschein, On the approximability of Dodgson and Young elections, Artificial Intelligence 187–188 (2012) 31–51.
[63] D. Marx, Parameterized complexity and approximation algorithms, The Computer Journal 51 (1) (2008) 60–78.
[64] M. Fellows, S. Gaspers, F. Rosamond, Parameterizing by the number of numbers, in: V. Raman, S. Saurabh (Eds.), Proceedings of the 5th International Symposium on Parameterized and Exact Computation, in: Lecture Notes in Computer Science #6478, Springer-Verlag, 2010, pp. 123–134.