A Computational Complexity Study of Various Types of Electoral Control, Cloning, and Bribery

Inaugural-Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Marc Neveling

aus Wuppertal

Düsseldorf, September 2021

aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Berichterstatter:

1. Herr Prof. Dr. Jörg Rothe

2. Frau Jun.-Prof. Dr. Dorothea Baumeister

Tag der mündlichen Prüfung: 20. September 2021

Abstract

This thesis deals with computational social choice which combines computational complexity theory, one of the most important areas of theoretical computer science, with social choice theory which is highly relevant to economists, politicians, and other figures involved in decision-making processes. Although being a fairly young research field, springing to life in the early 1990s, computational social choice has established itself as one of the central pillars of artificial intelligence and multi-agent systems research.

The central objects of interest for computational social choice are elections, which model decision-making processes where preferences of different agents or voters over candidates have to be aggregated into a final decision, and voting rules, which specify methods of how to aggregate those preferences. Naturally, all parties involved in an election, e.g., the agents, the organizing chair or even an outside agent, might have an interest to influence the outcome of an election. So-called election tampering attempts take many different forms: The agents may submit insincere preferences (i.e., strategic voting or election manipulation), the organizing chair might alter the structure of the election (i.e., electoral control) or an outside agent might want to bribe agents to change their votes to her liking (i.e., bribery). In this thesis we investigate, from a computational complexity perspective, to what degree elections evaluated by certain voting rules can be influenced by those types of election tampering attempts.

Firstly, we study electoral control for the Borda Count which is one of the oldest and most important voting rules. We consider different types of electoral control such as adding or deleting candidates or voters and in particular electoral control by partitioning the candidates or voters into two groups. Furthermore, we consider so-called online electoral control in which candidates or voters appear one after another in the election and the chair may decide only in the moment of appearance to exert some control action. We find that Borda is rather resistant against electoral control by proving NP-hardness of several control problems.

Secondly, we study replacement control which is a special kind of electoral control in which candidates or voters that are removed from the election need to be replaced by, as of yet, unregistered candidates or voters. We find that the complexity of replacement control problems usually follows the complexity of the corresponding classical control problems. Furthermore, we fill gaps in the literature regarding the complexity of the classical electoral control problems regarding adding or deleting candidates or voters.

Thirdly, we consider multiwinner elections in which we seek to elect a fixed-sized set of candidates, a committee, instead of single candidates. We devise a model and define several decision problems to model electoral control by cloning of candidates. A candidate is cloned by adding a new candidate to the election that is very similar to the original candidate. We study the introduced model for cloning candidates in multiwinner elections for several popular multiwinner voting rules and find a wide range of complexity results.

The last contribution of this thesis deals with shift bribery, which is a special kind of bribery in which only one special candidate may be moved forwards or backwards in the voters' preferences. We study shift bribery for iterative voting rules that decide the outcome of an election in several rounds. We find that iterative voting rules are generally very resistant to shift bribery. In contrast to non-iterative voting rules, for which several examples of vulnerability against shift bribery can be found in the literature, shift bribery is NP-hard for all of our considered iterative voting rules.

CONTENTS

1 Introduction **1**

2 Background **5**
2.1 Computational Complexity 5
2.2 Voting 12

3 Control Complexity in Borda Elections **29**
3.1 Summary 29
3.2 Publication – Neveling and Rothe [120] 29
3.3 Personal Contribution 30

4 Towards Completing the Puzzle: Complexity of Control **31**
4.1 Summary 31
4.2 Publication – Erdélyi, Neveling, Reger, Rothe, Yang and Zorn [58] 32
4.3 Personal Contribution 32

5 The Complexity of Cloning Candidates in Multiwinner Elections **33**
5.1 Summary 33
5.2 Publication – Neveling and Rothe [119] 34
5.3 Personal Contribution 34

6 Complexity of Shift Bribery for Iterative Voting Rules **35**
6.1 Summary 35
6.2 Publication – Maushagen, Neveling, Rothe, and Selker [106] 36
6.3 Personal Contribution 36

7 Conclusions **37**

Bibliography **39**

CHAPTER 1

INTRODUCTION

Collective decision-making, the act of aggregating individual preferences of a group of individuals into a final decision, is important in almost every social aspect of life ranging from politics over economics to everyday activities like choosing where to go on vacation. In the wake of digitalization a multitude of additional settings became important including multi-agent systems, meta-search engines or recommendation systems for online multimedia platforms like Youtube or Netflix. In each of those settings there is a set of candidates or alternatives from which we would like to choose one and a set of agents, which we will call voters, with preferences over the candidates. Depending on the specific settings candidates might be politicians to be elected, bills to be ratified, applicants to company positions or objects to be chosen from. The agents might be registered voters, jury or committee members, or even processes running on servers.

The most common way to aggregate individual preferences and come to a collective decision is to run an election: collect the preferences of the voters and use an aggregation procedure, which we will call voting rules, to determine the winning candidate. But, choosing a good voting rule for the job at hand is more intricate than at first glance. Firstly, it is important to consider in what form the voters' preferences are given. There may be ordinal preferences meaning voters order the candidates linearly according to their liking or there may be cardinal preferences in which the voters assign each candidate points or even a mixture of both. Still, even if we focus on one type of preferences, for example ordinal preferences, there is a multitude of possible voting rules we could use to aggregate those preferences. As early as the age of ancient Greece elections were used to elect representatives or settle disputes but the scientific study of elections did not start until the late 18th century when the Marquis de Condorcet [33] started a research field called social choice theory by applying mathematics to voting theory and rigorously formalizing elections and voting rules. With this, general theorems or statements can be deduced and voting rules can be characterized which hopefully helps us choose the right voting rule for the right task.

One of the many famous results of Condorcet's work is the Condorcet paradox. Consider the following linear preferences of three voters over the candidates a, b and c: Voter 1 prefers a to b to c, voter 2 prefers b to c to a, and voter 3 prefers c to a to b. Whichever candidate we choose as the winner there is always another candidate who is preferred by two of the three voters (e.g., if a is chosen as the winner, voter 2 and 3 both prefer candidate c to a) implying that it might not be possible to find a satisfying outcome for an election. Condorcet's preferred method of voting, thus called the "Condorcet method", chooses the candidate as the winner of an election that defeats all other candidates in pairwise comparison. Other famous theorems resulting from this "golden age" of social choice theory are the Condorcet jury theorem [33], the median voter theorem [18], and May's theorem [110]. But the most famous result is certainly due to Arrow [2], who later received a Nobel Memorial Prize in Economic Sciences together with John Hicks. In his so-called impossibility theorem it was shown that no voting rule, which accepts votes that rank all candidates, can satisfy three reasonable criteria

simultaneously:[1] (1) If every voter prefers some candidate a over some candidate b, then the voting rule cannot choose b over a, (2) if the voting rule prefers a to b then this is still the case even if candidates other than a and b are removed from the election, and (3) no single voter should be able to decide the outcome. The theorem implies that there does not exist a "perfect" voting rule.

Another method of voting was proposed by Jean-Charles de Borda to elect the members of the French Academy of Science in 1770 [20]. In the so-called "Borda method" candidates score points from every voter depending on where they are placed in the voter's preference. Condorcet and Borda famously argued whose voting method is better with Borda once defending his voting method by proclaiming "My scheme is intended only for honest men" [141]. He is thereby implying that when using his voting rule a voter can benefit by casting a dishonest vote, which is also called strategic voting. Strategic voting should ideally be discouraged since it would give the dishonest voters more influence over the election outcome as the other, honest voters. Alas, Allan Gibbard [78] and Mark Satterthwaite [142] showed independently from each other that a voting rule that is non-dictatorial, meaning that there is no single voter who determines the outcome, and non-imposing, meaning every candidate can possibly win in some election, is necessarily manipulable by strategic voting.[2] Since a fair voting rule should always be non-dictatorial and non-imposing, strategic voting is always possible. This notion of reasonable voting rules being manipulable is further reinforced by the Duggan-Schwartz Theorem [48]. Their theorem deals with non-resolute voting rules which means that they are choosing not a single candidate as the winner but a subset of candidates (i.e., candidates may tie for the win). Duggan and Schwartz show that a non-resolute voting rule that is anonymous (i.e., all voters are treated the same), non-imposing, and where there may be voters whose top ranked candidate is not in the set of winners is necessarily manipulable by strategic voting. The third property is reasonable to assume for a voting rule in order to have a meaningful set of winners since the winning set of a voting rule that includes all top ranked candidates in the set of winners would always be very large.

The celebrated Gibbard-Satterthwaite impossibility theorem inspired John Bartholdi III, Craig Tovey, and Michael Trick in the late 1980s to a series of papers applying computational complexity theory to social choice theory launching a research field called computational social choice that sits at the intersection of economics and computer science. Their approach to combat strategic voters was to choose a voting rule for which it is intractable to decide if an election can be manipulated by strategic voting. Then, under the condition that the election is large enough which is true in most multi-agent settings, it might take a strategic voter too long to decide if the election can be manipulated or to even compute a successful strategic vote [37]. Remarkably, intractability is seen as a positive property for this use case whereas it mostly seen negatively in computational complexity theory (i.e., the problem at hand cannot always be solved quickly). This notion is in many ways similar to how computational complexity was successfully used in cryptography to investigate the vulnerability of cryptosystems to attacks. Ideally, a cryptosystem should be hard to break, which can be achieved if it takes a long time for an attacker to break it [135].

The approach used to combat strategic voting can be extended to other forms of election tampering. Besides voters, other entities involved in elections could try to influence the outcome of an election. An election's chair organizing the election could be interested in steering the election into a certain direction without directly submitting a vote but instead by altering the structure of the election. We call this form of election tampering "electoral control" and it appears in the real world as, e.g., voter

[1] To be precise, we also need to require that there are at least three candidates and the votes are unrestricted in their structure.

[2] Again, we tacitly assume that there are at least three candidates and the votes are not restricted in some way.

disenfranchisement [154], cloning of candidates [150], or Gerrymandering [61, 91]. For voter disenfranchisement regulations are introduced to prevent groups of voters from voting that would vote contrary to the chair's liking. Felony disenfranchisement is the most common form of voter disenfranchisement in which people with criminal convictions are excluded from voting. In the US it is believed that felony disenfranchisement heavily influenced the presidential election in 2000 [105]. By cloning a candidate, i.e., introducing additional candidates to the election that are similar to an already participating candidate, the chair can split up the support of the cloned candidate. This type of electoral control is also known as the "strategic candidacy problem" [49]. Gerrymandering is used in district-based elections in which voters are partitioned into districts usually constrained by their geographical location. The voters then elect a representative for their district and the representatives of all districts elect the overall winner in a separate election. By manipulating the district borders the chair can reduce the impact votes of a certain group of voters have on the overall election result. This technique was first used by Massachusetts governor Elbridge Gerry in 1812 by creating a salamander shaped district coining the term "Gerrymandering" [61, 91].

Another way to tamper with an election is bribery. Now, an external agent, who is not part of the election, is bribing voters to change their vote in some way that favors the external agent's own preference. Bribery problems come in many different flavors depending on which voters can be bribed and how their votes can be changed [70]. Apart from the obvious example of a malicious briber trying to influence an election, a whole range of real world scenarios can be modeled by bribery problems. The most common scenarios are political campaign management and as a robustness measure for election results.[3] The former is related to bribery in that the campaign manager "bribes" the voters to change their votes by running, e.g., targeted ad campaigns. A special kind of bribery in which only the favorite candidate of the external agent can be moved forward in the votes models a type of ethical campaign management in which the campaign manager is not allowed to apply smear tactics to damage the standing of other candidates. The robustness of an election result is important in the following way. If the winner of the election can be dethroned by only a few changes to the votes, then the supposed winner might be incorrect due to vote counting errors or manipulation attempts.

Beyond election tampering, computational social choice is concerned with winner determination in elections—in particular, with possible and necessary winners [99]—winner prediction [35], and iterative elections [112]. Another sub-field deals with multiwinner elections that seek to elect a fixed-sized set of candidates, a committee, and can be used for, e.g., shortlisting or parliamentary elections [71]. Over time other research fields were integrated and adapted to computational social choice to address settings such as meta-search engines [50, 98], information extraction [145], planning [55], automated scheduling [81], collaborative filtering [128], computational linguistics [124], kidney exchanges [134], and assignments of students to schools [79]. Very recently, special attention was given to participatory budgeting [6, 13, 30] which is concerned with the question of how a budget can be allocated to different projects depending on the preferences of voters.

In this thesis we will extend the pioneering work of Bartholdi, Tovey, and Trick focusing on the line of research concerning electoral control and bribery.

[3] Another way to measure the robustness of election results is the margin of victory [155].

Outline

In Chapter 2 we will provide the background for the following chapters including preliminary definitions and theorems of voting theory and computational complexity theory, and a brief survey of previous related works in computational social choice. In Chapter 3 we investigate how resistant the Borda voting rule is against the classical electoral control defined by Bartholdi, Tovey, and Trick [10] and Hemaspaandra, Hemaspaandra, and Rothe [83]. In Chapter 4 we study electoral control once again solving open problems to try to complete the puzzle of the computational complexity of electoral control for the most popular voting rules. In particular, we study replacement control for which we may only remove a candidate or voter from the election if, in turn, we add an unregistered candidate or voter, respectively. In Chapter 5 we devise a new model, in the context of multiwinner elections, for electoral control by cloning, which is the act of adding candidates to an election that are similar (i.e., clones) to an already participating candidate, and study the computational complexity of this model for various multiwinner voting rules. In Chapter 6 we study shift bribery, which is a special kind of bribery in which only the position of a special candidate can be improved or worsened. For this type of bribery we study how resistant iterative voting rules, which elect the winner of an election in multiple rounds, are against it. Lastly, in Chapter 7 we summarize the previous chapters and provide starting points for future research.

CHAPTER 2

BACKGROUND

In this chapter we will provide background information necessary for the following chapters including definitions, notation, and some important prior work.

For a comprehensive overview of computational social choice see the recent book by Brandt et al. [23], especially the book chapters therein by Conitzer and Walsh [37] and Faliszewski and Rothe [70] concerning manipulation, electoral control, and bribery, and the book by Rothe [136]. A more concise introduction to computational social choice is provided by the surveys of Chevaleyre [31], Faliszewski and Procaccia [69], and Faliszewski, Hemaspaandra, and Hemaspaandra [64]. Additionally, see the very recent survey by Lang [99] of elections with incomplete knowledge. For an introduction to multiwinner elections see the book chapter by Kilgour [95] and the book chapter by Faliszewski et al. [71] surveys recent research from the computational complexity perspective. The books by Arora and Barak [1], Papadimitriou [126], and Rothe [135] deal with complexity theory (see Tovey's tutorial [151] for an introduction). For parameterized complexity theory see the books by Cygan et al. [39], Downey and Fellows [47], Flum and Grohe [76], and Niedermeier [122].

2.1 Computational Complexity

Our main tools for analyzing elections and classifying voting rules originate in a research field called computational complexity theory started by the seminal work of Hartmanis and Stearns [80] in 1965 who use the abstract computational model of Turing machines defined by Alan Turing [152]. Computational complexity theory deals with the question of whether different computational tasks can be solved "efficiently". By efficiently we mean the amount of resources (usually the time it takes to finish or the space that is used) required to complete the computation. Computational tasks are formalized as decision problems[1] that consist of the problem's name, given information (i.e., the input), and a question about the input. For example, the important decision problem SAT [77] is defined as follows.

SATISFIABILITY (SAT)	
Input:	A boolean formula ϕ with a set of (boolean) variables X and a set of clauses K over X.
Question:	Is there a satisfying truth assignment to the variables of ϕ?

[1] Note that there are many different kinds of problems besides decision problems such as optimization problems, search problems, sort problems or counting problems but in this thesis we will only deal with decision problems. Therefore, when we speak of "problems" we always mean "decision problems".

A specific case of a decision problem is called an *instance* (i.e., for SAT an instance would be a specific boolean formula). We call an instance I of a decision problem A a *yes-instance* if the answer to the question of the problem for this instance is "yes" and a *no-instance* otherwise. Sometimes it is useful to think of the decision problem as the set of all yes-instances of the problem so $I \in A$ if and only if I is a yes-instance of A (e.g., $\text{SAT} = \{\phi \mid \phi \text{ is a satisfiable boolean formula}\}$).

Example 2.1 (Instances of decision problems)**.** Consider the instance (X,K) of SAT with $X = \{x_1, x_2\}$ and $K = \{K_1, K_2\}$ with $K_1 = (x_1 \vee \overline{x_2})$ and $K_2 = (\overline{x_1} \vee \overline{x_2})$. Can we find a truth assignment to x_1 (i.e., x_1 is set to *true* or *false*) and x_2 (i.e., x_2 is set to *true* or *false*) such that both K_1 and K_2 evaluate to *true*? Here, the answer is *yes* since we can set x_1 to *true* and x_2 to *false* to satisfy both clauses. Therefore, (X,K) is a yes-instance of SAT. If K would contain a third clause $K_3 = (x_2)$, we would have a no-instance since we cannot satisfy K_1, K_2, and K_3 at the same time.

Note that the input to a decision problem can be of many forms. In addition to boolean formulas they may have strings, integers, graphs, elections, etc. as inputs. To be able to do computations on different kinds of input we assume—on a lower level—that inputs are encoded in some way, usually as strings over $\{0,1\}$ called the *binary encoding*.

A (deterministic) *algorithm* that solves a decision problem takes any input or instance to the problem (in binary encoding), does some basic computational steps, and outputs the answer to the question of the problem, therefore *deciding* if the instance belongs to the problem or not. Assuming the Church-Turing Thesis [32, 152], that any real-world algorithm can be simulated by Turing machines, holds true we use Turing machines as the computational model to represent algorithms (although, for the sake of readability, we will use a more descriptive language to define algorithms in this thesis).

The type of algorithms defined above are deterministic meaning the steps the algorithm takes for an input are singular and predetermined leading to a *computation path* in which every node is a state of the algorithm. In contrast, a nondeterministic algorithm may choose from several possible steps on how to proceed with the computation on the current state. In particular, the algorithm makes every possible decision (i.e., which computational step to do next) simultaneously building a *computation tree* instead of a path. Then, the nondeterministic algorithm accepts the input if there is at least one path in its computation tree that accepts. Note that some paths on the computations tree of a nondeterministic algorithm may be infinitely long in contrast to deterministic algorithms in which every computation stops eventually. Therefore, nondeterministic algorithms can only accept decision problems and never decide them like deterministic algorithms. Due to the fact that there are no modern computers who can run nondeterministic algorithms (yet) they would have to be simulated by deterministically running through every path in the computation tree if used in practice which is very inefficient. For a deterministic algorithm we define its (worst-case) runtime as the maximum number of steps the algorithm takes over all possible inputs. For a nondeterministic algorithm its (worst-case) runtime is the maximum number of steps of the shortest accepting path over all inputs that the algorithm accepts.[2] Runtimes are given as functions depending on the size of the input that describe the growth of the computational effort of the algorithm with increasing size of the input. For those function we are only interested in the fastest growing factor, called the *runtime bound*, as slower growing factors and constants become irrelevant for larger input sizes (e.g., the values of the functions $f(x) = 2x+3$ and $g(x) = 3x$ for very large x are very similar but much smaller in comparison to values of $h(x) = 2^x$). That means we are interested in the asymptotic behavior,

[2] There are other complexity measures such as space that we will not discuss here.

the asymptotic bounds, of algorithms when the size of the input increases. In this thesis, the most important runtime functions are polynomial and exponential functions. A polynomial function p is defined as $p(x) = c_k x^k + c_{k-1} x^{k-1} + \cdots + c_1 x + c_0$ for some constant k and $k+1$ constants $c_0, \ldots, c_k$, and an exponential function f is defined as $f(x) = 2^{p(x)}$ for some polynomial function p. We say that an algorithm has a polynomial-time runtime if its runtime functions is in $O(p(x))$ for some polynomial function p and an exponential-time runtime if its runtime function is in $O(f(x))$ for some exponential function f.[3]

Complexity Classes

Computational complexity theory aims to group similarly complex problems together into so-called *complexity classes*. The most important complexity classes are P, containing problems that can be solved in deterministic polynomial time, and NP, containing problems that can be solved in nondeterministic polynomial time. In order to show that a decision problem belongs to P we need to find a deterministic polynomial-time algorithm that solves the problem and for NP-membership the algorithm only needs to be nondeterministic.

Example 2.2 (SAT is in NP)**.** To show that SAT is in NP, a nondeterministic algorithm could (nondeterministically) guess a truth assignment for the variables of a given instance of SAT and then check (in polynomial time) if it satisfies the given formula of the instance.

It is widely believed that P represents the class of problems that can be efficiently solved (problems belonging to P are also sometimes called *tractable*) although a polynomial runtime with a large exponent is still very slow for large inputs. But it turns out that many natural problems in P actually have algorithms with polynomial runtime with only small exponents so this assumption seems reasonable. Obviously, it holds that $\mathrm{P} \subseteq \mathrm{NP}$ since every deterministic algorithm is also nondeterministic with a computation tree that only consists of one path. We already mentioned that nondeterministic algorithms can be simulated by deterministic algorithms with an exponential blow-up in the runtime. The question of whether there exists a deterministic polynomial-time algorithm for every nondeterministic polynomial-time algorithm is formalized as the famous open $\mathrm{P} = \mathrm{NP}$ problem [38]. It is widely believed that $\mathrm{P} \neq \mathrm{NP}$, i.e., there exist problems in NP for which no deterministic polynomial-time algorithm exists and we also require this assumption in this thesis. Another complexity class is coNP which contains the complements of problems that are in NP (i.e., let A be a decision problem and I be an instance of A, then I is in the complement of A if and only if $I \notin A$). We mention in passing that there are other complexity classes that contain problems who are (allegedly) even harder than problems in P, NP or coNP. Together they form the so-called *polynomial hierarchy* introduced by Meyer and Stockmeyer [115] and Stockmeyer [149] but we will not further discuss those complexity classes here.

Showing that a decision problem belongs to a complexity class is in some sense showing an upper bound of the complexity of that problem since then we can follow that the problem can be solved at least as fast as the most difficult problems in that complexity class. In this thesis we aim to find upper bounds of P or NP. In contrast, showing a lower bound of the complexity for a problem is much more

[3] We assume the reader to be familiar with the Big O notation. Roughly, for two functions f and g, f is in $O(g)$ if there is some input to f and g after which f does not grow faster than g (i.e., f is upper bounded by g barring constant factors and finitely many exceptions).

complicated as we would need to show that there does not exist an algorithm with a specific runtime that solves the problem. The notion of reducibility which we will discuss in the following section enables us to proof such lower bounds for decision problems.

Reducibility and Complexity Lower Bounds

We say that a decision problem A (polynomial-time many-one)[4] reduces to another decision problem B (formally, $A \leq_m^p B$) if we can construct in polynomial time from each instance I of A an instance I' of B such that I is a yes-instance of A if and only if I' is a yes-instance of B. Intuitively, if we can reduce A to B, then A is at least as hard to solve as B. Then, a decision problem A is $\leq_m^p$-hard for a complexity class $\mathscr{C}$ (or simply $\mathscr{C}$-hard) if $B \leq_m^p A$ for every $B \in \mathscr{C}$. Furthermore, a decision problem A is $\leq_m^p$-complete for a complexity class $\mathscr{C}$ (or simply $\mathscr{C}$-complete) if A is $\mathscr{C}$-hard and $A \in \mathscr{C}$. The notion of hardness for a complexity class intuitively means that a decision problem is at least as hard to solve as the hardest problems of that complexity class and completeness for a complexity class even implies that the problem is one of the hardest problems in this class. Even still, in order to show hardness with this basic definition we would need to reduce every one of (possibly) infinitely many problems in the complexity class to the problem we want to show hardness for. Luckily, due to the transitivity of $\leq_m^p$ (i.e., for three problems A, B, and C, it holds that if $A \leq_m^p B$ and $B \leq_m^p C$, then $A \leq_m^p C$), if a decision problem B is $\mathscr{C}$-hard for a complexity class $\mathscr{C}$ and $B \leq_m^p A$ for another decision problem A, then A is $\mathscr{C}$-hard as well. Thus, given a $\mathscr{C}$-hard problem we can show $\mathscr{C}$-hardness of another problem by reducing from the $\mathscr{C}$-hard problem to it. Cook [38] showed that SAT is NP-hard opening up the possibility to show NP-hardness (and even NP-completeness) for many other problems by reducing from it. Karp [93] then showed for several natural problems that they are NP-complete. Since P contains the decision problem that can be solved in polynomial time, showing NP-hardness and assuming $\mathrm{P} \neq \mathrm{NP}$ implies that the problem is not solvable efficiently or that it is intractable. This makes the $\mathrm{P} \neq \mathrm{NP}$ question central to computational complexity theory as important natural problems (e.g., the problem of creating mathematical proofs) are NP-hard and a collapsing of both complexity classes would mean that they are efficiently solvable. Interestingly, showing for only one NP-complete problem that it is in P would immediately show P-membership of all other NP-complete problems as well and, thus, showing $\mathrm{P} = \mathrm{NP}$. Note that NP-hardness of a problem does not mean that specific instances of the problem are hard to solve as the size of a specific instance is predetermined and, therefore, can be solved in constant time. Rather, the NP-hardness of a problem means that, unless $\mathrm{P} = \mathrm{NP}$, as instances of the problem increase in size the time to solve those instances grows exponentially.

We can also use reducibility to show upper bounds. Since P, NP, and coNP are closed under $\leq_m^p$-reducibility, for a complexity class $\mathscr{C} \in \{\mathrm{P}, \mathrm{NP}, \mathrm{coNP}\}$ and two decision problems, A and B, with $A \leq_m^p B$ and $B \in \mathscr{C}$ it follows that $A \in \mathscr{C}$.

Parameterized Complexity Theory

Showing NP-hardness is not the end of research but merely the beginning. There are several approaches on how to deal with the NP-hardness of a decision problem and gain more insight into what

[4] There are many other notions of reducibility but in this thesis we will only use polynomial-time many-one reducibility so we will simply call it "reducibility".

makes the problem hard to solve. Apart from average-case analysis, finding approximate solutions, or using heuristics, studying the parameterized complexity of a problem has seen much attention recently.

The general idea of parameterized complexity theory is to choose some part of the input to an NP-hard decision problem as the parameter and then design an algorithm that runs in polynomial time if the parameter is small or even a constant (it can only run in exponential time in general as the problem is NP-hard, unless P = NP) or proof that such an algorithm probably does not exist. To this end, we turn a decision problem into a *parameterized decision problem* by taking some part of the input as the parameter. An instance of such a parameterized decision problem is a pair (I, p) with I being an instance of the "not-parameterized" problem and p being some part of I.

Example 2.3 (Parameterization of VERTEX COVER)**.** Consider the following decision problem.

VERTEX COVER	
Input:	An undirected graph $G = (V, E)$ with n vertices and m edges and an integer k.
Question:	Does there exist a set $V' \subseteq V$ of at most k vertices of G such that V' covers the edges of G (i.e., for each $\{v_1, v_2\} \in E$, $v_1 \in V'$ or $v_2 \in V'$)?

The obvious parameters of VERTEX COVER are the integer k, the number of vertices n, or the number of edges m. We can also combine parameters by adding them, e.g., we could choose the parameter $k+n$. We may also study parameters that are given by the structure of the instance. For problems that deal with graphs, like VERTEX COVER, it is common to choose the maximum degree of the graph or the treewidth of the graph as parameters.

A parameterized decision problem is in FPT (i.e., it is fixed-parameter tractable) if there exists an algorithm that solves it in $O(f(p) \cdot |I|^{O(1)})$ time for some computable function f. If p is small or a constant, then $|I|^{O(1)}$ is the fastest growing factor of the runtime of an FPT-algorithm which turns the runtime polynomial. The complexity class XP consists of problems that can be solved in $O(f(p) \cdot |I|^{g(p)})$ time for some computable functions f and g. Notice that if the parameter is a constant, then the problem can be solved in polynomial time. Therefore, problems in this complexity class are often called *slice-wise* polynomial-time solvable as instances (I, p) of a parameterized problem for some fixed p can be seen as a *slice* of the problem. In practice XP-algorithms often times do not perform very well as the constant parameter might be relatively large which results in a high degree polynomial as the runtime. In contrast, the degree of the polynomial in the runtime of an FPT-algorithm must be independent of the parameter.

Example 2.4 (VERTEX COVER parameterized by k is in FPT)**.** Since VERTEX COVER is NP-complete (see, e.g., Garey and Johnson [77]) we expect that there is no polynomial-time algorithm that solves the problem. The trivial brute-force algorithm that iterates through all $\binom{n}{k}$ possible sets of vertices of size k and checks whether it is a vertex cover obviously needs exponential time in the worst case ($O(n^k)$ time to be precise) to solve the problem. Notice that if k is a constant, the algorithm runs in polynomial time. Therefore, VERTEX COVER parameterized by k is in XP. On the other hand, the algorithm above does not satisfy the requirements to be a FPT-algorithm. But, we can improve the trivial algorithm in the following way. Firstly, we can remove any vertex from the graph that has no neighbor since we cannot cover any edges with such a vertex. Afterwards, notice that, given an

instance (G,k) of VERTEX COVER, if a vertex v of G has at least $k+1$ neighbors, this vertex needs to be in a vertex cover for otherwise the cover would have to include all at least $k+1$ neighbors of v to cover all of v's incident edges. Therefore, we can immediately remove such a vertex (including its incident edges) from the graph and decrease k by one. So, after those two reduction steps all remaining vertices have at most k neighbors but at least one and we can reject the instance if there are more than k^2 edges since every vertex can cover at most k edges. Otherwise, it follows that there are at most $2k^2$ vertices that were not removed in the reduction step and we can brute-force over all (at most) $\binom{2k^2}{k}$ sets of vertices of size k to check whether all edges are covered. This algorithm has a worst case runtime of $O(2^k k^{2k} \cdot m + nm)$ and, therefore, runs in FPT-time if k is the parameter.

The technique shown in Example 2.4 to first reduce the size of an instance which can then be solved in FPT-time is also called *kernelization*. Another technique to show FPT-membership uses integer linear programs. An *integer linear program* (ILP) consists of p variables and a set of m linear inequalities over the variables. Then, the goal of the ILP is to choose values for the variables such that the inequalities are satisfied. Formally, we define the INTEGER LINEAR PROGRAMMING FEASIBILITY problem as follows. We are given a constraint matrix $A \in \mathbb{Z}^{m \times p}$ and a bias vector $b \in \mathbb{Z}^m$. The question is whether there exists a variable vector $x \in \mathbb{Z}^p$ such that $Ax \leq b$. INTEGER LINEAR PROGRAMMING FEASIBILITY is known to be NP-complete but the following theorem was proven by Lenstra [101].

Theorem 2.1 (Lenstra's theorem [101])**.** *An* INTEGER LINEAR PROGRAMMING FEASIBILITY *instance of size L with p variables can be solved in time $O(p^{2.5p+o(p)} \cdot L)$.*

It follows that INTEGER LINEAR PROGRAMMING FEASIBILITY is in FPT if parameterized by p, i.e., the number of variables. Therefore, if we manage to express a parameterized problem as an ILP in which the number of variables is only bounded by the parameter, we can solve this ILP (and subsequently the problem) in FPT-time by using Lenstra's theorem. We illustrate the technique by the following example.

Example 2.5 (An ILP for VERTEX COVER)**.** Given an instance (G,k) of VERTEX COVER, the ILP has a boolean variable x_i for every $v_i \in V$ (i.e., $x_i \in \{0,1\}$ for $1 \leq i \leq n$). If a variable x_i is set to 1, this corresponds to vertex v_i being in the vertex cover. Furthermore, the ILP consists of the following constraints.

$$x_i \geq 0 \qquad \text{for every } v_i \in V \tag{2.1}$$

$$x_i + x_j \geq 1 \qquad \text{for every } \{v_i, v_j\} \in E \tag{2.2}$$

$$\sum_{i=1}^{n} x_i \leq k \tag{2.3}$$

Constraint (2.2) ensures that every edge of the graph is covered and constraint (2.3) ensures that the vertex cover contains at most k vertices. Due to Theorem 2.1 and the fact that we have n variables, we can solve the ILP in FPT-time if n is the parameter and, therefore, have shown that VERTEX COVER is in FPT for this parameter. Note that parameterizing VERTEX COVER by n is not all that useful since the size of the instance is directly related to n (i.e., $k \leq n$ and $m \leq \frac{n(n-1)}{2}$). Intuitively, that means if we assume n to be small, then instances are small as well and the result that those instances are fast to solve is unsurprising.

The ILP technique described above was successfully used in computational social choice to show FPT-membership of problems dealing with winner determination [9], bribery [24, 29, 45], control [66], possible winner [15, 26], and lobbying [25]. We will also use this technique in Chapter 5 to show FPT-membership.

Similarly to NP-hardness we can define a notion of hardness for parameterized problems as well showing (under a separation assumption similar to P $\neq$ NP) that there is no FPT-algorithm for a parameterized problem. First, we need another set of parameterized complexity classes, the so-called W hierarchy. We omit the details of defining it formally as it is out of scope of this thesis (see the book by Downey and Fellows [47] for the formal definitions). The most important fact is that for every $t \geq 1$ there is a parameterized complexity class W[t] and it holds that FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq$ $\cdots \subseteq$ XP. Showing that a parameterized problem is W[t]-hard for some $t \geq 1$ and assuming that FPT $\neq$ W[1] then prevents the problem from being in FPT. For showing W[t]-hardness we need to extend the notion of reducibility to parameterized problems. We say a parameterized decision problem A, with parameter p, reduces to another parameterized decision problem B, with parameter r, if we can construct for each instance (I,p) of A in $O(f(p)\cdot|I|^{O(1)})$ time, for some computable function f, an instance (I',p') with $p' \leq g(p)$ for some computable function g such that (I,p) is a yes-instance of A if and only if (I',p') is a yes-instance of B. The main difference to "not-parameterized" reductions is that we need the constructed parameter for the instance of the target problem to be *exclusively* bounded by some function of the parameter of the original instance. With this notion of reducibility we can reduce some W[t]-hard problem to some other parameterized problem to show the W[t]-hardness of the latter problem. Although the formal definition of the W hierarchy is quite technical involving combinatorial circuits, for W[1] and W[2], there are natural problems with natural parameterizations that are hard for one of those classes and from which we can reduce to show hardness for other parameterized problems. For W[1]-hardness the following problem can be used.

MULTICOLORED CLIQUE	
Input:	Given an undirected graph $G = (V,E)$, an integer f, and a partition of V into f sets $W_1,\ldots,W_f$.
Question:	Does there exist a clique $X \subseteq V$ (i.e., the induced subgraph of G restricted to X is complete) that contains exactly one vertex of every set W_i with $1 \leq i \leq f$?

MULTICOLORED CLIQUE is W[1]-hard if parameterized by f [46]. For W[2]-hardness a central problem is SET COVER which is defined as follows.

SET COVER	
Input:	Given a set $X = \{x_1,...,x_m\}$, a family $\mathscr{S} = \{S_1,\ldots,S_n\}$ of subsets of X, and an integer k.
Question:	Does there exist a cover of X of size at most k, i.e., a subfamily $\mathscr{S}' \subseteq \mathscr{S}$ with $\lvert\mathscr{S}'\rvert \leq k$ such that $\bigcup_{S_j \in \mathscr{S}'} S_j = X$?

If SET COVER is parameterized by k, it is W[2]-hard [46]. We note in passing that hardness for some class of the W hierarchy does not, in general, imply NP-hardness since a parameterized reduction allows the construction to be done in FPT-time with respect to the parameter which might be exponential-time with respect to the size of the input.

Lastly, we will discuss the parameterized complexity class para-NP which sits above XP (unless P = NP). A parameterized decision problem is para-NP-hard if it is NP-hard for some constant value of the parameter. Intuitively, if some slice of a parameterized problem is intractable, then it cannot belong to XP (unless P = NP) since this would imply that all slices are tractable. Interestingly, para-NP bridges the gap between parameterized complexity and classical complexity as it can be shown that FPT = para-NP if and only if P = NP [76].

2.2 Voting

An *election* is defined as a pair (C,V) with C being a finite set of *candidates* and V being a multiset of the voters' preferences over C, sometimes referred to as the *preference profile*. Typically, voters express their preference (i.e., the vote or the ballot they cast) as a linear order $\succ$ over the candidates in C with the following three properties.

1. Completeness: For every pair of candidates $c,d \in C$, we have $c \succ d$ or $d \succ c$;

2. Transitivity: For every triplet of candidates $c,d,e \in C$, if $c \succ d$ and $d \succ e$, it follows that $c \succ e$;

3. Antisymmetry: For every pair of candidates $c,d \in C$, if $c \succ d$, then $d \succ c$ does not hold.

For example, given a set of candidates $C = \{a,b,c,d\}$ a voter that prefers b to a, a to d, and d to c would have the preference $b \succ a \succ d \succ c$ (sometimes we omit the $\succ$ symbols and write the preference as a string $b\ a\ d\ c$). Note that the first and third property imply that the voter is sure for every pair of candidate which one the voter prefers over the other, i.e., the preferences are *strict*. In the (computational) social choice literature it is sometimes allowed that the voter may be indifferent of candidates (i.e., we drop the completeness property) but in this thesis we will always assume strict preferences. Apart from those *ordinal preferences* voters' preferences may be *cardinal* which means that each voter assigns each candidate a number of points. A special type of cardinal preferences are *approval-based preferences* in which a voter can only assign the values 0 or 1 to a candidate corresponding to the voter, respectively, disapproving or approving the candidate. Then, the vote is simply given as the subset of approved candidates. In some cases voters may even have preferences that are a mixture of both cardinal and ordinal preferences. In the following, we will assume that voters' preferences are ordinal and explicitly mention it in the few cases where it is not the case.

The outcome of an election is determined by a *voting rule*[5] that is a mapping which assigns every possible election a subset of the set of candidates which form the winners of the election. For some voting rule $\mathscr{E}$ and an election (C,V) we call a candidate that is part of the set of winners of the election under $\mathscr{E}$ an $\mathscr{E}$-winner of (C,V). A resolute voting rule, in which always only a single candidate can be a winner, is also known as a *social choice function*. If we are looking for a complete linear order over the candidates as an outcome of an election, we speak of *social welfare functions*.

We will now define the studied voting rules.

[5] In the literature, voting rules are often referred to as social choice correspondences, voting protocols, or voting systems.

Positional Scoring Rules

An important class of voting rules are (positional) scoring rules. Let m be the number of candidates. Then, scoring rules use a so-called *scoring vector* $(\alpha_1, \alpha_2, \ldots, \alpha_m)$ with each α_i, $1 \leq i \leq m$, being a positive integer called a *score value* and $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_m$ (i.e., the monotonicity of the score values) to determine the score of each candidate (i.e., the candidate in position i in a vote gains α_i points and the points are summed up over all votes) and the candidate(s) with the highest score win(s). Therefore, each scoring rule is defined by a series of scoring vectors; one scoring vector for each possible number of candidates. To represent this infinite series of vectors succinctly Betzler and Dorn [14] defined the class of *pure scoring rules* in which we can obtain the scoring vector of size m by inserting an additional score value somewhere into the scoring vector of size $m-1$ maintaining the monotonicity of the scoring vector as described above. As Hemaspaandra, Hemaspaandra, and Schnoor [89] observe we can also assume that $\alpha_m = 0$ and that there is no integer which divides all score values which restricts the class of pure scoring rules only slightly.

We focus on the following (pure) scoring rules.

k-approval: The first k score values are 1 and all others are 0. 1-approval is also known as **plurality**.

k-veto: The last k score values are 0 and all others are 1. 1-veto is simply called **veto**.

Borda (Count): Let m be the number of candidates. Then, we have for each i, $1 \leq i \leq m$, that $\alpha_i = m - i$.

We can also define *iterative* variants of scoring rules in which the winner(s) are determined in several rounds.

Hare: Uses plurality scores to eliminate, in each round, the candidates with the lowest score until all remaining candidates have the same score which are proclaimed winners of the election. This voting rule is often known as single transferable voting (STV) but we will use the name STV for the multiwinner variant below.

Coombs: Works the same as Hare but uses veto scores.

Baldwin: Works the same as Hare but uses Borda scores.

Nanson: Uses Borda scores but eliminates all candidates that have less then the average Borda score which is defined as $(m-1)\frac{n}{2}$ with m being the number of candidates and n the number of voters.

Plurality/Veto with runoff: We always have only two rounds. In the first round only the candidates with the highest plurality/veto score proceed to the second round, except when there is a unique winner in which case the candidates with the highest and second highest plurality/veto score proceed to the next round. In the second round plurality/veto scores are used to determine the winner(s).

Iterated plurality/veto: All candidates are eliminated that do not have the highest plurality/veto score.

Notice that in all iterative voting rules we eliminate all candidates if there is a tie in some round. Sometimes a tie-breaking rule (e.g., a linear order over the candidates deciding which candidate is eliminated first if a tie occurs) is used to break ties instead. Then, only one candidate is eliminated in each round.

Lastly, we define the fallback voting rule which is a hybrid between positional scoring rules and approval-based voting rules.

Fallback: Instead of linear orders we assume that voters' preferences are given as a set of approved candidates and disapproved candidates while the former set is ordered linearly as well. For example, given a set of candidates $\{a,b,c,d,e\}$ a voter might have the set $\{a,b,c\}$ as approved candidates, the set $\{d,e\}$ as disapproved candidates, and orders the former set as $b \succ c \succ a$. Then this voter's vote would be written as $b \succ c \succ a \mid \{d,e\}$. We call a vote a level-i approval for some candidate c if c is in the first i positions of the voter's approved set of candidates. Furthermore, we call a candidate a level-i winner if the candidate is in the first i positions of approved sets of candidates in at least half of the voters' preferences. Then, the fallback winners are those candidates that are level-i winners for the smallest i and have the highest number of level-i approvals. If there are no such candidates, then fallback chooses the candidates with the most (overall) approvals as the winners.

Condorcet Extensions

The following voting rules rely on pairwise comparisons of the candidates. For an election (C,V) and two candidates $c,d \in C$, let $N_{(C,V)}(c,d)$ be the number of voters whose preferences rank c in front of d. Condorcet is one of the oldest and most prominent voting rules but has the downside that there may not be a winner at all. Therefore, Copeland and maximin try to imitate Condorcet by always choosing a Condorcet winner if there is one and provide a nonempty set of winners otherwise.

Condorcet: The Condorcet-winner is a candidate c who beats all other candidates in direct comparison (i.e., $N_{(C,V)}(c,d) > N_{(C,V)}(d,c)$ for all candidates $d \in C \setminus \{c\}$).

Copeland: The Copeland$^\alpha$ score with $0 \leq \alpha \leq 1$ and for a candidate $c \in C$ is defined as

$$|\{d \in C \setminus \{c\} \mid N_{(C,V)}(c,d) > N_{(C,V)}(d,c)\}| + \alpha|\{d \in C \setminus \{c\} \mid N_{(C,V)}(c,d) = N_{(C,V)}(d,c)\}|.$$

Intuitively, c gains a point for each candidate that c beats in direct comparison and α points for each tie. Then, the Copeland$^\alpha$ winners are the candidates with the highest Copeland$^\alpha$ score. Copeland$^\alpha$ with $\alpha = \frac{1}{2}$ is referred to as **Copeland**.

Maximin: The maximin score of a candidate c is defined as $\min_{d \in C \setminus \{c\}} N_{(C,V)}(c,d)$. Then, the candidates with the highest maximin score are the winners.

Range Voting

For the last two voting rules we assume cardinal preferences. That means a voter's preference is a point vector $v \in \{0,1,\ldots,k\}^m$ of size m and describes the amount of points a voter assigns to every candidate. (We assume here that the candidates are ordered lexicographically such that the i-th component of the vector corresponds to the i-th candidate according to this ordering.) The number k is the

maximum number of points a voter can give to a single candidate. For an election (C,V), if k is fixed and every voters gives at most k points to a candidate we call (C,V) a *k-range election*. Note that in a k-range election it is not required that all voters give 0 or k points to some candidate which in reality is very unlikely as voters tend to maximize the points given to their favorite candidate and minimize the points given to their most despised candidate. We will later see how votes are normalized to display this behavior.

Range Voting: Given a k-range election, we simply sum up the points each candidate is given by the voters and the candidates with the highest score are k-range voting winners. 1-range voting is commonly known as **approval voting**.

Normalized Range Voting: Given a k-range election, we first normalize each voter's point vector as follows. For a candidate $c \in C$ and a voter $v \in V$, let s be the points the candidate is given by this voter and let $s_{\min}$ and $s_{\max}$ be the minimal and maximal points given to any candidate by this voter. Then, the normalized score that c is given by voter v is $\frac{k(s-s_{\min})}{s_{\max}-s_{\min}}$. We can assume that $s_{\min}$ and $s_{\max}$ are not equal for otherwise the voter would be indifferent of every candidate. Similarly to range voting, we will then sum up the normalized points for each candidate and the candidates with the highest normalized score win.

For all of the voting rules above the outcome of an election can be computed in polynomial time. Voting rules for which winner determination[6] is NP-hard (e.g., Kemeny [94] and Dodgson [44]) will not be discussed here but some of them will be defined later in the context of multiwinner elections.

Manipulating Elections

We will now explore how different actors in an election may be able to influence its outcome. We start with insincere voters which is often known as strategic voting. Consider the following example.

Example 2.6 (Manipulation)**.** Let $C = \{a,b,c,d\}$ and consider the following four voters in V.

$$v_1 : c \succ d \succ b \succ a$$
$$v_2 : a \succ d \succ c \succ b$$
$$v_3 : d \succ c \succ b \succ a$$
$$v_4 : a \succ d \succ c \succ b$$

The Borda scores of the candidates in (C,V) are as follows. Candidate a has 6 points, b has 2 points, c has 7 points, and d has 9 points. So, d is the unique Borda-winner of the election. Now consider the first voter v_1 and assume that before she casts her vote she finds out how the other voters vote and that together with her (honest) vote her favorite candidate c does not win. Then, she might be tempted to change her vote to $c \succ b \succ a \succ d$ which would lead to c being tied for the win together with a and d.

[6] The problem $\mathscr{E}$-WINNER DETERMINATION for a singlewinner voting rule $\mathscr{E}$ is defined by the input that consists of an election (C,V) and candidate $c \in C$ and the question of whether c is an $\mathscr{E}$-winner of (C,V). On page 28 we will define the problem for multiwinner voting rules as well, which is slightly different than this variant for singlewinner voting rules.

Strategic voting with only one manipulating voter is formalized as the $\mathscr{E}$-MANIPULATION problem for some voting rule $\mathscr{E}$ which was first defined by Bartholdi, Tovey, and Trick [8] who studied the complexity of this problem for various voting rules.

$\mathscr{E}$-MANIPULATION	
Input:	An election (C,V), an additional voter v (the strategic voter), and a distinguished candidate c.
Question:	Is there a vote that v can cast such that c is an $\mathscr{E}$-winner of the election $(C,V\cup\{v\})$?

Note that in contrast to Example 2.6 the honest vote of the strategic voter is not given in the input and we simply ask if there is any vote v can cast so that c wins which might as well be her honest vote. Example 2.6 illustrates that a voter can actually benefit from casting a strategic vote instead of an honest one.

For most voting rules it seems that MANIPULATION with only a single manipulator is easy since Bartholdi, Tovey, and Trick [8] provided a simple greedy algorithm that solves MANIPULATION for many common voting rules. Surprisingly, they also found a natural voting rule for which MANIPULATION is NP-hard, namely second-level Copeland which is the Copeland voting rule with a tie-breaking mechanism. Later on, the NP-hardness of MANIPULATION was also shown for Hare with single-candidate elimination by Bartholdi and Orlin [7] and for a voting rule called Ranked Pairs by Xia et al. [158].

If there is more than one strategic voter and they work together, this is called coalitional manipulation and was formalized by Conitzer, Sandholm, and Lang [36] for so-called weighted elections. In a weighted election (C,V,w) in addition to the set of candidates C and the multiset of the voters' preferences V we are given a weight function $w : V \to \mathbb{N}$ that assigns every voter a positive integer. Although elections with weighted votes are violating the democratic principle that all votes should be weighted equal in many settings this is not the case. For example, the countries in the European Union are weighted and in decision processes within a company the shareholders' votes are weighted depending on how many shares each shareholder holds. Up until now we have always assumed that the manipulating actor has a favorite candidate that she wants to make a winner of the election but she may also have a despised candidate which she wants to prevent from winning. Conitzer, Sandholm, and Lang [36] described those two notions as constructive manipulation and destructive manipulation letting them define the following decision problems as their central problems to study.

$\mathscr{E}$-CONSTRUCTIVE-COALITIONAL-WEIGHTED-MANIPULATION ($\mathscr{E}$-CCWM)	
Input:	A weighted election (C,V,w), a coalition of manipulators V' with weights w', and a distinguished candidate c.
Question:	Are there votes the manipulators in V' can cast such that c is an $\mathscr{E}$-winner of the weighted election $(C,V\cup V',w\cup w')$?

The destructive variant ($\mathscr{E}$-DCWM) has the same input and asks whether there are votes the manipulators of V' can cast such that c is *not* an $\mathscr{E}$-winner of the weighted election $(C,V\cup V',w\cup w')$. Those two problems were studied for a variety of voting rules by Conitzer, Sandholm, and Lang [36] especially under the aspect of how many manipulators are needed until coalitional manipulation becomes intractable for a voting rule. Continuing this line of research, Hemaspaandra and Hemaspaandra [82]

settled the complexity of weighted manipulation for the class of all scoring rules by showing that weighted manipulation is intractable for a scoring rule if and only if it satisfies the "diversity of dislike" property (i.e., the score value of the second best and worst candidate in a vote are different).

In contrast, much less is known of the unweighted variant (i.e., each weight is 1) of these problems: Faliszewski, Hemaspaandra, and Schnoor [68] showed that coalitional manipulation is intractable for Copeland voting while it is tractable with only one manipulator; Davies et al. [41] and Betzler, Niedermeier, and Woeginger [16] independently showed that coalitional manipulation is intractable for Borda; and other results were shown by Xia et al. [157, 158] and Narodytska and Walsh [116].

Electoral Control

Besides manipulation, Bartholdi, Tovey, and Trick initiated the study of electoral control in 1992 [10]. Instead of manipulation attempts by voters electoral control deals with election tampering attempts by the election chair that organizes the election. The chair can influence the structural parts of the election which is the set of candidates or the voters that participate in the election and might even be able to influence the election process by holding subelections.

Example 2.7 (Electoral control)**.** Consider the election of Example 2.6 again evaluated with Borda. An election chair who would like c to win might choose to remove the candidate d from the election which would change the original election as follows.

Original election $(\{a,b,c,d\},V)$	Controlled election $(\{a,b,c\},V)$
$v_1 : c \succ d \succ b \succ a$	$v_1 : c \succ b \succ a$
$v_2 : a \succ d \succ c \succ b$	$v_2 : a \succ c \succ b$
$v_3 : d \succ c \succ b \succ a$	$v_3 : c \succ b \succ a$
$v_4 : a \succ d \succ c \succ b$	$v_4 : a \succ c \succ b$

In the controlled election, a has 4 points, b has 2 points, and c has 6 points turning c into the unique winner of the election while d won the original election uniquely.

Bartholdi, Tovey, and Trick [10] defined eleven different decision problems dealing with various kinds of election tampering by the chair which were doubled to 22 problems by Hemaspaandra, Hemaspaandra, and Rothe [83] who defined the destructive variants of the original electoral control problems. The problems concerned with altering the set of candidates and the multiset of voters' preferences can be conveniently combined to the following problem called multimode control problem which was first defined by Faliszewski, Hemaspaandra, and Hemaspaandra [65].

$\mathscr{E}$-CONSTRUCTIVE-MULTIMODE-CONTROL	
Input:	An election $(C \cup D, V \cup W)$ with C and D being disjoint sets of, respectively, registered and unregistered candidates and V and W being disjoint multisets of, respectively, preferences of registered and unregistered voters, four nonnegative integers $\ell_{DC}, \ell_{AC}, \ell_{DV}$, and ℓ_{AV}, and a distinguished candidate $c \in C$.
Question:	Are there subsets $C' \subseteq C \setminus \{c\}, D' \subseteq D, V' \subseteq V$, and $W' \subseteq W$ with $\|C'\| \leq \ell_{DC}, \|D'\| \leq \ell_{AC}, \|V'\| \leq \ell_{DV}$, and $\|W'\| \leq \ell_{AV}$ such that c is an $\mathscr{E}$-winner of the election $((C \setminus C') \cup D', (V \setminus V') \cup W')$?

Then, we obtain the special cases

$\mathscr{E}$-Constructive-Control-by-Adding-Candidates ($\mathscr{E}$-CCAC) by setting $\ell_{DC} = \ell_{AV} = \ell_{DV} = 0$ and $W = \emptyset$;

$\mathscr{E}$-Constructive-Control-by-Adding-an-Unlimited-Number-of-Candidates ($\mathscr{E}$-CCAUC) by setting $\ell_{DC} = \ell_{AV} = \ell_{DV} = 0$, $\ell_{AC} = |D|$, and $W = \emptyset$;

$\mathscr{E}$-Constructive-Control-by-Deleting-Candidates ($\mathscr{E}$-CCDC) by setting $\ell_{AC} = \ell_{AV} = \ell_{DV} = 0$ and $D = W = \emptyset$;

$\mathscr{E}$-Constructive-Control-by-Adding-Voters ($\mathscr{E}$-CCAV) by setting $\ell_{AC} = \ell_{DC} = \ell_{DV} = 0$ and $D = \emptyset$; and

$\mathscr{E}$-Constructive-Control-by-Deleting-Voters ($\mathscr{E}$-CCDV) by setting $\ell_{AC} = \ell_{DC} = \ell_{AV} = 0$ and $D = W = \emptyset$.

We will study those problems for various voting rules in Chapter 4.

The second set of problems deals with partitioning the set of candidates or multiset of voters' preferences. Then, one or two subelections are run before the overall winners are decided by a final election with a reduced set of candidates. Note that in an election with a reduced set of candidates the votes are always masked down to the participating candidates (see Example 2.8 below). Bartholdi, Tovey, and Trick [10] also considered two notions of tie-breaking in the subelections. The first one is called *ties-promote* (TP) in which all tied candidates proceed to the final and the second one is called *ties-eliminate* (TE) in which only a unique winner of a subelection proceeds to final and all candidates are eliminated if there is a tie for the win. We can now define the decision problems. Given an election (C,V) and a distinguished candidate $c \in C$, $\mathscr{E}$-CONSTRUCTIVE-CONTROL-BY-RUNOFF-PARTITION-OF-CANDIDATES ($\mathscr{E}$-CCRPC) asks whether we can partition C into two disjoint subsets C_1 and C_2 such that c is an $\mathscr{E}$-winner of the two stage election in which the winners of the first stage (sub)elections (C_1,V) and (C_2,V) (with either ties-eliminate or ties-promote tie-breaking) proceed to a second and final runoff election in which the overall winners are determined. Then, we can define the following variants that have the same input as $\mathscr{E}$-CCRPC but ask slightly different questions.

- For $\mathscr{E}$-CONSTRUCTIVE-CONTROL-BY-PARTITION-OF-CANDIDATES ($\mathscr{E}$-CCPC) we ask the same question as for $\mathscr{E}$-CCRPC except we only run one subelection in the first stage, (C_1,V), and the candidates from C_2 get a bye to the final election, and

- $\mathscr{E}$-CONSTRUCTIVE-CONTROL-BY-PARTITION-OF-VOTERS ($\mathscr{E}$-CCPV) asks the question of whether there is a partition of V into two disjoint subsets, V_1 and V_2, such that c is an $\mathscr{E}$-winner of the two stage election in which the winners of the first stage (sub)elections (C,V_1) and (C,V_2) (with either ties-eliminate or ties-promote tie-breaking) proceed to a second and final runoff election with all voters in V in which the overall winners are determined.

Again, we obtain the destructive versions of the above problems by asking if we can make sure that c is not a winner of the election and replacing "Constructive" with "Destructive" in the problem names. To indicate which tie-breaking mechanism is used, we append "TE" for ties-eliminate tie-breaking or "TP" for ties-promote ties-breaking to the problem names. We illustrate how electoral control by partitioning the set of candidates works with the following example.

Example 2.8 (Control by runoff partition of candidates)**.** We will use ties-eliminate tie-breaking in this example. Consider the election of Example 2.6 which d wins uniquely if the election is evaluated by Borda. If we want do prevent d from winning (i.e., destructive control), we might choose to partition the candidate set C into $C_1 = \{a,d\}$ and $C_2 = \{b,c\}$. Then the first stage contains the following subelections with the votes being masked down to the respective reduced sets of candidates.

First subelection $(\{a,d\},V)$	Second subelection $(\{b,c\},V)$
$v_1 : d \succ a$	$v_1 : c \succ b$
$v_2 : a \succ d$	$v_2 : c \succ b$
$v_3 : d \succ a$	$v_3 : c \succ b$
$v_4 : a \succ d$	$v_4 : c \succ b$

If we again use Borda to evaluate the subelections, a and d tie for the win in the first subelection and are eliminated due to ties-eliminate tie-breaking. Therefore, we have achieved our goal to prevent d from winning. For completeness, c beats b in the second subelection and proceeds to the final election which is won by c as well since she is the only candidate still standing.

In contrast to manipulation we might not be able to influence the election outcome by some type of electoral control using some voting rule. For example, assume we try to make some candidate a winner of an election evaluated with Condorcet by adding additional candidates. Then, either the candidate is already a winner of the election or she is beaten by some candidate in pairwise comparison which we cannot change by adding additional candidates to the election. In this case we would call the voting rule *immune* against this type of electoral control. Otherwise we call the voting rule *susceptible* to this type of electoral control and further investigate the computational complexity of the associated decision problem. If we can show that the decision problem is solvable in polynomial time, we call the voting rule *vulnerable* to this type of electoral control or if we can show that the decision problem is NP-hard, we call the voting rule *resistant* against this type of electoral control.

Electoral control has since been studied extensively for a variety of voting rules [56, 59, 66, 108, 109, 114, 127]. Currently, the voting rules with the most resistances against electoral control types are fallback (see the work of Erdélyi et al. [56]) and normalized range voting (see the work of Menton [114]) who are only vulnerable to two of the 22 types. Although Hemaspaandra, Hemaspaandra, and Rothe [84] constructed a hybrid voting rule that is resistant against all types while still being

computationally easy to compute there is still no "natural" voting rule that is resistant to all 22 types. In Section 3 we will continue the study of classical types of electoral control for the Borda Count.

Another type of electoral control that was not part of the set of classical control types but related to control by adding candidates is control by cloning of candidates. The notion of cloning candidates in elections was first studied by Tideman [150] as the so-called *independence of clones* property of voting rules and was later formalized as a decision problem by Elkind, Faliszewski, and Slinko [53]. They defined the action of cloning candidates as a size-m vector of nonnegative integers $(k_1,\ldots,k_m)$ with m being the number of candidates and some arbitrary (e.g., lexicographic) ordering of the candidates. Each entry k_i, $1 \leq i \leq m$, in the vector with $k_i > 0$ means that the i-th candidate of the election is replaced by k_i clones $c_i^{(1)},\ldots,c_i^{(k_i)}$. If $k_i = 0$ for some i with $1 \leq i \leq m$, then c_i stays in the election and no clone for this candidate is added. Notice that this definition of cloning candidates is slightly different than in the work of Elkind, Faliszewski, and Slinko [53] in that we allow candidates to not be cloned (Elkind, Faliszewski, and Slinko [53] replace every candidate by at least one clone) which seems more natural and does not restrict the model. Given an election $E = (C,V)$ with $C = \{c_1,\ldots,c_m\}$ and a vector $K = (k_1,\ldots,k_m)$ of nonnegative integers, a *cloned election* $E^* = (C^*,V^*)$ via K is derived from E by the set of candidates

$$C^* = \left(C \setminus \bigcup_{k_i \in K, k_i > 0} \{c_i\}\right) \cup \left(\bigcup_{k_i \in K, k_i > 0} \{c_i^{(1)},\ldots,c_i^{(k_i)}\}\right)$$

and for every vote $v_j \in V$ there is a vote $v_j^* \in V^*$ which is a (complete) linear order over C^* such that for every pair of candidates $c_i, c_j \in C$ it holds for v_j that $c_i \succ c_j$ if and only if $c_i' \succ c_j'$ in v_j^* with $c_i' = c_i$ if $k_i = 0$ or for every $c_i' \in \{c_i^{(1)},\ldots,c_i^{(k_i)}\}$ otherwise, and $c_j' = c_j$ if $k_j = 0$ or for every $c_j' \in \{c_j^{(1)},\ldots,c_j^{(k_j)}\}$ otherwise. We illustrate the notion of cloned elections with the following example.

Example 2.9 (Cloned elections)**.** Let $E = (C,V)$ be an election with $C = \{c_1,c_2,c_3\}$ and V consisting of two voters with preferences $v_1 : c_1 \succ c_2 \succ c_3$ and $v_2 : c_2 \succ c_1 \succ c_3$. Consider the vector $K = (0,2,0)$ which means that c_1 and c_3 remain in the election but c_2 is replaced by two clones $c_2^{(1)}$ and $c_2^{(2)}$ yielding $C^* = \{c_1,c_2^{(1)},c_2^{(2)},c_3\}$. Regarding the voters, v_1 might be extended to $v_1^{(1)} : c_1 \succ c_2^{(1)} \succ c_2^{(2)} \succ c_3$ or $v_1^{(2)} : c_1 \succ c_2^{(2)} \succ c_2^{(1)} \succ c_3$ and v_2 to $v_2^{(1)} : c_2^{(1)} \succ c_2^{(2)} \succ c_1 \succ c_3$ or $v_2^{(2)} : c_2^{(2)} \succ c_2^{(1)} \succ c_1 \succ c_3$. Thus, there are four possible cloned elections of E via K (i.e., $(C^*,\{v_1^{(1)},v_2^{(1)}\})$, $(C^*,\{v_1^{(1)},v_2^{(2)}\})$, $(C^*,\{v_1^{(2)},v_2^{(1)}\})$, and $(C^*,\{v_1^{(2)},v_2^{(2)}\})$).

Notice that there are several possible cloned elections depending on how the clones of a single candidate are ordered against each other. Therefore, the following decision problem is defined for some $q \in \{0^+\} \cup (0,1]$ that describes the *probability of success* that we need to reach. That means that q is the fraction of all possible cloned elections in which the distinguished candidate needs to be a winner. The special case $q = 0^+$ means that we need only one cloned election in which the distinguished candidate is a winner in order to be successful. To decide to what degree we can clone candidates we are given a cost function $\rho : [m] \times [t] \to \mathbb{N} \cup \{+\infty\}$ for some integer $t > 1$. Then, $\rho(i,j)$ defines the cost of adding the jth clone of the ith candidate to the election. Since adding the first clone of a candidate only replaces the original candidate we require $\rho(i,1) = 0$ for every $i, 1 \leq i \leq m$.

$\mathscr{E}$-q-CLONING	
Input:	An election (C,V), a distinguished candidate $c \in C$, a positive integer $t > 1$, a cost function $\rho : [m] \times [t] \to \mathbb{N} \cup \{+\infty\}$, and a budget B.
Question:	Is there a vector of nonnegative integers $K = (k_1, \ldots, k_m)$ with $\sum_{k_i \in K} \sum_{j=2}^{k_i} \rho(i,j) \leq B$ such that c (or some clone of c) is an $\mathscr{E}$-winner of a cloned election of (C,V) via K with probability q?

Elkind, Faliszewski, and Slinko [53] also considered two special cases with the cost functions that have $\rho(i,j) = 0$ for all i, $1 \leq i \leq m$, and $j \in \mathbb{N}$ which is called ZERO COST (ZC) and $\rho(i,j) = 1$ for all i, $1 \leq i \leq m$ and $j \geq 2$ which is called UNIT COST (UC).

Bribery

In contrast to manipulation and electoral control, the notion of bribery was introduced to computational social choice only much later by Faliszewski, Hemaspaandra, and Hemaspaandra [63]. Bribery assumes that there is an outside agent that tries to influence an election by bribing the voters to change their vote to the outside agent's preference.

Example 2.10 (Bribery)**.** Again, consider the election of Example 2.6 evaluated with Borda. If we want the candidate c to be the winner, we can bribe the second voter to change her vote to $c \succ a \succ d \succ b$ which would lead to c scoring 9 points while d scores 8 points, a scores 5 points, and b scores 3 points.

Similarly to the previous section we will define a very general bribery problem (see the book chapter by Faliszewski and Rothe [70]) that captures the different flavors of bribery by Faliszewski, Hemaspaandra, and Hemaspaandra [63] and by Elkind, Faliszewski, and Slinko [52].

$\mathscr{E}$-CONSTRUCTIVE-PRICED-BRIBERY	
Input:	An election (C,V) with m candidates and n voters, a list of price functions $(\rho_1, \ldots, \rho_n)$ such that for each i, $1 \leq i \leq n$, and each possible linear order υ over C, $\rho_i(\upsilon)$ is the price to pay so that voter i changes her vote to υ, a distinguished candidate c, and a positive integer B.
Question:	Can we bribe the voters in V with a budget of B such that c becomes an $\mathscr{E}$-winner of the resulting election?

Then, we can define the other bribery problems by restricting the range of price functions the voters may have. The problems $\mathscr{E}$-CONSTRUCTIVE-BRIBERY and $\mathscr{E}$-CONSTRUCTIVE-$BRIBERY defined by Faliszewski, Hemaspaandra, and Hemaspaandra [63] have so-called *discrete* and *$discrete* price functions, respectively. We call a price function ρ_i discrete if $\rho_i(\upsilon_i) = 0$ with υ_i being the preference order of voter i and $\rho_i(\upsilon) = 1$ for every preference order $\upsilon \neq \upsilon_i$ (intuitively, the briber pays nothing for not bribing and can freely change the preference of a voter for unit cost). A $discrete price function ρ_i is defined similarly except that we have $\rho_i(\upsilon) = c_i$ for every preference order $\upsilon \neq \upsilon_i$ with c_i being some constant (i.e., the price of voter i to be bribed which may vary for different voters). The third type of price functions are *swap-bribery* price functions which were first introduced by

Faliszewski et al. [66] for irrational voters[7] and later studied by Elkind, Faliszewski, and Slinko [52] for voters with linear preference orders. Formally, a swap-bribery price function ρ_i is defined by a constant $c_i^{\{x,y\}}$ for each pair of candidates $x, y \in C$ such that for each preference order υ, $\rho_i(\upsilon)$ is the sum of all constants $c_i^{\{x,y\}}$ for which the candidates $x, y \in C$ are in opposite order in υ and υ_i, the preference order of voter i (intuitively, the briber pays a voter to swap two candidates in her preference order). SWAP-BRIBERY turned out to be NP-hard for most of the voting rules considered by Elkind, Faliszewski, and Slinko [52] so they also studied a natural special case called *shift bribery*. For $\mathscr{E}$-SHIFT-BRIBERY swap-bribery price functions are used with the restriction that for each pair of candidates $x, y \in C \setminus \{c\}$, with c being the distinguished candidate, we have $c_i^{\{x,y\}} = B + 1$ (intuitively, the briber can only shift the distinguished candidate forwards or backwards in the voters' preference orders). Swap bribery and shift bribery have natural applications in practice as they model campaign management. A campaign manager for a specific candidate might try to improve her candidate's chances of winning by running ads that target specific groups of voters and make them change their opinion of the ordering of candidates. Shift bribery models a more ethical approach to campaign management as only the position of the distinguished candidate (i.e., the candidate for which the campaign is managed) may be altered by campaign management actions such as ads. Due to this very natural application, shift bribery has been thoroughly studied since its introduction. Schlotter, Faliszewski, and Elkind [143] studied shift bribery for approval-like voting rules; Bredereck et al. [24] studied shift bribery for several classes of price functions; Kaczmarczyk and Faliszewski [92] studied destructive shift bribery; Bredereck et al. [29] studied shift bribery in the context of multiwinner elections; and Bredereck et al. [28] studied a combinatorial variant of shift bribery in which one bribe action causes changes to the preferences of multiple voters. In Section 6 we will extend the study of shift bribery to the iterative voting rules defined above.

Each of the above bribery problems can also be defined for (a) weighted elections which will be denoted by adding "Weighted" to the problem names and (b) with a destructive goal which will be denoted by replacing "Constructive" with "Destructive" in the problem names. Destructive bribery is especially interesting as it can measure the *robustness* of an election result (see the work of Xia [104] for a more detailed discussion): If the winner of an election can be dethroned by only a few changes to the election, the current winner might be wrong due to vote counting errors or even raise the suspicion of election manipulation. The robustness of election results (in the context of multiwinner elections) was also studied by Bredereck et al. [27] although their method of investigating robustness is not directly related to bribery. Furthermore, Dey, Misra, and Narahari [125] studied frugal bribery in which a voter can only be bribed if the change to her vote improves the election result for this voter with respect to her preference; Faliszewski [62] studied so-called nonuniform bribery which is a model of bribery for (k,b)-elections which is a special type of elections in which voters submit their preferences by allocating k points to the candidates while never giving a candidate more than b points; and Erdélyi, Hemaspaandra, and Hemaspaandra [57] studied bribery under the assumption that the voting rule used to evaluate the election is not fixed, i.e., there is uncertainty about which voting rule is used.

Notice that in all decision problems that we have defined above the goal is to make the distinguished candidate a winner of the election which means that the distinguished candidate does not need to beat every candidate but at least tie them. This is called the *nonunique-winner model*. In contrast, we can ask for the distinguished candidate to be the unique winner which is then called the *unique-winner*

[7] In contrast to (rational) voters having linear preference orders, an irrational voters may have cycles in her preference order. For example, given a set of three candidates $\{a,b,c\}$ an irrational voter may prefer a to b, b to c, and c to a.

model. The former is more common in the computational social choice literature which is why we use this winner model as well. We will later see that the choice of the winner model is not only a matter of taste but there might even be a change in complexity for some decision problems when the winner model is changed.

Possible and Necessary Winners

Up until now we required the voters to have complete preferences over the candidates. In practice, this is very rarely the case: The ballots of voters are kept secret until the election is over and some voting rules, such as plurality, do not require complete preferences. Moreover, complete preferences might not even be desirable: Does a voter really know who she prefers of every pair of candidates or are most of them simply ordered randomly or, even worse, lexicographically? Regarding "unrealistic" complete preferences, one could argue that if some type of election tampering is hard with full information, it is at least as hard with only partial information. Still, it makes sense to study elections with partial information.

We can define *partial* preferences from complete preferences by dropping the completeness property (i.e., a partial preference is a linear order over the candidates that is transitive and antisymmetric). Usually, a partial preference is defined by a set of pairwise comparisons of the form $c_i \succ c_j$. Then, a partial preference profile is a multiset of the voters' partial preferences. A complete preference v' over a set of candidates C *extends* a partial preference v over C if for all $c_i, c_j \in C$ it holds that if $c_i \succ c_j$ in v, then $c_i \succ c_j$ in v'. We call a multiset of complete preferences $\{v'_1, \ldots, v'_n\}$ an *extension* of a multiset of partial preferences $\{v_1, \ldots, v_n\}$ if for every i, $1 \leq i \leq n$, v'_i extends v_i.

We can now define the $\mathscr{E}$-POSSIBLE-WINNER and $\mathscr{E}$-NECESSARY-WINNER problems introduced by Konczak and Lang [97].

$\mathscr{E}$-POSSIBLE-WINNER	
Input:	An election (C,V) with a set of candidates C and a partial preference profile V and a distinguished candidate c.
Question:	Is there an extension V' of V to complete preferences such that c is an $\mathscr{E}$-winner of the election (C,V')?

$\mathscr{E}$-NECESSARY-WINNER is defined similarly but we ask whether c is an $\mathscr{E}$-winner of the election (C,V') *for all* extensions V' of V.

Both problems were further studied by Xia and Conitzer [156], Walsh [153], Pini et al. [130], Betzler, Hemmann, and Niedermeier [15], Betzler and Dorn [14], and Baumeister and Rothe [12]. Interestingly, $\mathscr{E}$-Possible-Winner generalizes the $\mathscr{E}$-CONSTRUCTIVE-COALITIONAL-MANIPULATION problem [97] and is itself a special case of $\mathscr{E}$-SWAP-BRIBERY [52].

Electoral Control in Sequential Elections

Another partial information model was introduced and studied by Hemaspaandra, Hemaspaandra, and Rothe [85, 86, 87, 88] in a series of papers concerning different kinds of election tampering attempts in sequential elections. We will define the so-called *online* models for electoral control [86, 87] in detail

and refer to the corresponding papers for the online models for manipulation [85] and bribery [88]. Later in Section 3 we will study the *online* models for electoral control for the Borda Count.

Online candidate control [86] models voting scenarios in which the candidates are added to the election (and evaluated against the already participating candidates by the voters) one after the other and the election chair may decide, only at the moment a candidate appears and never after that, to exert a control action (such as adding or deleting) on this candidate. The corresponding online control problems *online constructive control by deleting candidates* for a voting rule $\mathscr{E}$ (online-$\mathscr{E}$-CCDC), *online constructive control by adding candidates* for a voting rule $\mathscr{E}$ (online-$\mathscr{E}$-CCAC) and their destructive variants online-$\mathscr{E}$-DCDC and online-$\mathscr{E}$-DCAC capture such a *moment of decision* for the election chair. For online-$\mathscr{E}$-CCDC we are given the set of candidates C, the set of voters V (note that only in this section the voters' preferences are given separately later as they are not complete over the set of candidates), the election chair's ideal ranking σ over the candidates, the election chair's distinguished candidate $d \in C$, an order of the candidates describing in which order they appear in the election with a flag for each candidate saying who the current candidate is and which of the already revealed candidates were deleted, the voters' preferences over the still standing (i.e., already revealed but not deleted) candidates including the current candidate, and the number of deletions k that the election chair has left to use. Then we ask whether the election chair can make a decision about the current candidate (whether to delete her if possible or not) so that the chair has a *forced win* by which we mean that no matter what happens in the future (i.e., how not yet revealed candidates appear in the voters' preferences) the chair can make decisions on later revealed candidates with the information available at the time such that the distinguished candidate d or some candidate ranked higher than d according to the chair's ranking σ is an $\mathscr{E}$-winner of the election in which only the not-deleted candidates participate.

The following example illustrates how a moment of decision and a forced win work for online-$\mathscr{E}$-CCDC.

Example 2.11 (Online control by deleting candidates)**.** In this example we will use plurality as the voting rule. Consider the following instance of online-$\mathscr{E}$-CCDC.

- Let $C = \{a, c, d, e\}$ and $V = \{v_1, v_2\}$.
- The chair's ranking is $d \succ a \succ b \succ c$.
- The distinguished candidate is d (i.e., the chair succeeds only if d wins).
- The candidates' order of appearance is $d\ a\ b\ c$.
- No candidate has been deleted as of yet and the current candidate is a (i.e., d and a are already revealed).
- The voters preferences are $v_1 :\ d \succ a$ and $v_2 :\ a \succ d$ (note that b and c have not shown up yet and are therefore not included in the preferences).
- Finally, $k = 2$.

Now the chair has to decide whether the current candidate a must be removed or not in order to have a forced win (i.e., no matter how the not yet revealed candidates b and c appear in the voters' preferences there exist decisions about b and c such that d wins). Notice that if the chair decides to remove a, then there is only one removal left so either b or c must remain in the election. In the worst case the

candidate that cannot be deleted will be ranked above d in both preferences and therefore beats d. So, by removing a the chair does not have a forced win. If the chair decides to leave a in the election, both b and c can be removed from the election later so no matter how they actually appear in the preferences at future moments of decision they will not appear in the preferences after all candidates have been revealed. Since d wins the election $(\{a,d\},V)$, the chair has a forced win by not deleting a.

For online-$\mathscr{E}$-CCAC the input changes slightly: We now have a set of *registered* candidates that are certainly part of the election and a disjoint set of unregistered candidates that may be added to the election only at the moment of decision when such a candidate is revealed. The order of appearance of candidates is over the union of both sets and the rest stays the same (the flag for a candidate now indicates whether an already revealed, unregistered candidate has been added to the election by the election chair and the deletion limit k is now an addition limit).

For the destructive variants, online-$\mathscr{E}$-DCDC and online-$\mathscr{E}$-DCAC, the chair's goal is to make sure none of the candidates d or worse in their ideal ranking win after all decisions have been made. Regarding online-$\mathscr{E}$-DCDC the election chair might try to delete all candidates d or worse to win trivially so Hemaspaandra, Hemaspaandra, and Rothe [86] proposed two approaches to prevent this behavior. The first one is called the *non-hand-tied chair model* and lets the chair delete some but never all candidates d or worse. In contrast the *hand-tied chair model* prevents the election chair from deleting any candidates d or worse.

The **online voter control** model [87] assumes that the set of candidates that are part of the election is fixed but now the voters are revealed sequentially (with preferences over the full set of candidates) and susceptible to control actions by the election chair, again, only at the moment they are revealed. Before we define the control problems that were introduced by Hemaspaandra, Hemaspaandra, and Rothe [87] we define the general information that all of them have in common namely an *online voter control setting* (OVCS) given by (C,u,V,σ,d) which contains the candidate set C, the current voter u, an *election snapshot* $V=(V_{<u},u,V_{u<})$ with $V_{<u}$ being the set of voters that were revealed before u and $V_{u<}$ being the set of not-yet-revealed voters, the election chair's ideal ranking σ of the candidates, and a distinguished candidate d. Note that $V_{<u}$ and u have already cast their votes so their preference orders are known but $V_{<u}$ only specifies the order in which the not-yet-revealed voters cast their votes. Then, the question is whether the election chair can make a decision about the current voter (whether to exert the control action at hand if possible or not) to have a *forced win* (i.e., the election chair can reach their—constructive or destructive—goal by making future decisions about not-yet-revealed candidates with the—up to each point of decision—revealed information). As before, the constructive goal of the chair is to make the candidate d or some candidate that is ranked higher than d in their ideal ranking a winner of the election after all voters have shown up and all decisions about the voters have been made, and the destructive goal aims to prevent all candidates d or worse in the chair's ideal ranking from winning.

For *online control by deleting voters* (i.e., the problems online-E-CCDV and online-E-DCDV) in addition to an OVCS we are given a nonnegative integer k (the number of deletions the election chair has left to use) and for each voter of $V_{<u}$ a flag that says whether the voter was deleted or not. The election after the voting process includes all voters that were not deleted by the election chair.

For *online control by adding voters* (i.e., the problems online-E-CCAV and online-E-DCAV) the OVCS is, again, augmented by a nonnegative integer k which is the limit of additions the election chair may use and for each voter there is a flag that indicates whether the voter is unregistered (i.e.,

the chair can choose to add her or not) or registered (i.e., the voter is definitely in the election) and for each voter in $V_{<u}$ there is another flag indicating whether this voter was added to the election. The election after all voters have shown up then includes the registered voters and all unregistered voters that have been added by the election chair.

Lastly, for *online control by partition of voters* (i.e., the decision problems online-E-CCPV and online-E-DCPV) the chair partitions the set of voters by assigning each voter to the left or the right part of the partition after they are revealed. Then, after all voters have been revealed the election proceeds in two stages in which the winners of two subelections with each part of the partition determine the overall winners in a final runoff election with all voters. So, in addition to an OVCS we are given a flag for every voter in $V_{<u}$ that indicates whether a voter was assigned to the left part or the right part of the partition. Similarly to the classical control by partition problems we adopt either the ties-promote model or the ties-eliminate model to decide whether, respectively, all candidates or none of the candidates that are tied for the win in a subelection proceed.

Multiwinner Voting

Another branch of the computational social choice landscape is concerned with elections that have a fixed-sized set of candidates—a committee—as the election outcome. This type of elections are known as *multiwinner elections* (in the literature they are also sometimes called *committee elections*). The notion of multiwinner elections and committees was implicitly introduced by Fishburn [75] in the context of so-called *choice functions* although the committee size was not fixed then. Debord [43] and Felsenthal and Maoz [74] later introduced k-choice functions which always output a size-k committee.

In comparison to singlewinner elections, in which the most popular candidate should be the winner, for multiwinner elections there are several approaches of which committee might be considered the "best" winning committee for a given election. Depending on the specific application of multiwinner elections the properties a winning committee should have change fundamentally. Elkind et al. [51] distinguish between three kinds of multiwinner elections:

Excellence-Based Elections: (Used for short-listing candidates for awards or job positions.) The winning committee should contain the most popular or highest-rated candidates.

Selecting a Diverse Committee: (Used for choosing items to display on a storefront or offer to a group of people.) The chosen candidates should be as diverse as possible.

Proportional Representation: (Used for parliamentary elections.) We seek to choose candidates such that the different views of the voters are represented proportionally in the committee.

Under those aspects we must choose a (multiwinner) voting rule that delivers an appropriate committee for a given application. Categorizing and analyzing multiwinner voting rules under those aspects to be able to choose the right voting rule for the right task has been given much attention (see, e.g., the work of Elkind et al. [51], Aziz et al. [3], Kilgour, Brams, and Sanver [96], Faliszewski et al. [72], Skowron, Faliszewski, and Lang [146], and Skowron, Faliszewski, and Slinko [147]). Interestingly, an impossibility theorem similar to that of Gibbard and Satterthwaite for singlewinner voting rules [78, 142] can be formulated for multiwinner voting rules as well: Peters [129] showed that

no multiwinner voting rule can simultaneously satisfy a weak proportionality property[8] and a weak form of strategy-proofness. Multiwinner voting rules usually fall into one of three categories: *Committee Scoring Rules* [147], in which voters' preferences are given as linear orders, *Approval-Based Counting Rules* [95], in which voters submit a subset of approved candidates, and *Condorcet-Inspired Rules* [4, 144]. We focus on multiwinner voting rules of the first category and refer to the corresponding literature for definitions of the other two.

Formally, a multiwinner election (C,V,k) is defined by a (singlewinner) election (C,V) with the set of m candidates C and the preference profile V of n voters augmented with a nonnegative integer k which is the size of the committee that we seek to elect. Given the committee size k, a multiwinner voting rule $\mathscr{E}$ is a function mapping each multiwinner election (C,V,k) to a nonempty family of size-k subsets of C, the winning committees of (C,V,k) under $\mathscr{E}$. We will now define the multiwinner voting rules that we focus on.

Single transferable vote (STV): Given the quota $q = \lfloor \frac{n}{k+1} \rfloor + 1$, we choose candidates for a winning committee iteratively as follows. We compute plurality scores and if some candidate reaches the quota, we add her to the committee and remove q voters that vote for her. If no candidate reaches the quota, we remove the candidate with the lowest plurality score from the election (i.e., we remove her from all preference orders in the preference profile and the voters voting for this candidate now transfer their vote to the second highest candidate in their preference order).

An important issue is how ties are handled especially since we might need to break ties between voters in cases when a candidate has more than q voters voting for him but only q of them are removed. Conitzer, Rognlie, and Xia [34] devised a very fair tie-breaking scheme called *parallel-universes tiebreaking* (PUT) for which a committee is winning under STV if there exists a series of choices, breaking ties, such that the candidates of the committee are chosen by using STV. Sadly, using this tie-breaking method makes determining whether a committee is winning (see the corresponding decision problem below) intractable [34].

Single nontransferable vote (SNTV): Choose k candidates with highest 1-approval score.

Bloc: Choose k candidates with highest k-approval score.

k-Borda: Choose k candidates with highest Borda score.

$\mathscr{E}$-Chamberlin–Courant ($\mathscr{E}$-CC): Given a scoring rule $\mathscr{E}$, each committee is assigned a score by each voter that is the score under scoring rule $\mathscr{E}$ that the highest-ranked member of the committee in the voter's preference order would receive from the voter. The committee(s) with the highest overall score, summed up over all voters, are winning. We focus on **k-approval-CC** and **Borda-CC** which use k-approval and Borda scores, respectively.

For all $\mathscr{E}$-CC rules, the winner determination problem (defined below) is intractable [103, 132] but it is in FPT if parameterized by the number of candidates or voters [17].

Regarding computational considerations a central problem is the *winner determination* problem which was studied for various multiwinner voting rules by Aziz et al. [5], Procaccia, Rosenschein, and Zohar [131] and Baumeister, Dennisen, and Rey [40].

[8] Proportionality for multiwinner voting rules roughly means that a produced winning committee must represent the voters' preferences proportionally.

$\mathscr{E}$-WINNER DETERMINATION	
Input:	A multiwinner election (C,V,k) and a size-k committee $C' \subseteq C$.
Question:	Is C' a winning committee of (C,V,k) under $\mathscr{E}$?

The study of election tampering attempts in multiwinner elections was initiated by Meir et al. [113] focusing on strategic voting and proceeded by Aziz et al. [5], Obraztsova, Zick, and Elkind [123], and Baumeister, Dennisen, and Rey [40]. Bredereck et al. [27, 29] and Faliszewski et al. [73] studied bribery in multiwinner elections. In Section 5 we will extend the model of electoral control by cloning candidates for singlewinner elections that was introduced above to the multiwinner setting and study it for the multiwinner voting rules above.

CHAPTER 3

CONTROL COMPLEXITY IN BORDA ELECTIONS: SOLVING ALL OPEN CASES OF OFFLINE CONTROL AND SOME CASES OF ONLINE CONTROL

3.1 Summary

The Borda Count is one of the most important voting rules which finds applications not only in voting settings but also can be used for the allocation of indivisible goods [21] (for a thorough introduction to the field of fair division see, e.g., the book chapter by Lang and Rothe [100]) and hedonic games [138] (an introduction to hedonic games can be found in, e.g., the book chapter by Elkind and Rothe [54]). We first survey recent research in all three fields relating to Borda.

Then, we study electoral control for Borda. Especially the electoral control problems involving partitioning the set of candidates or voters were largely unexplored for Borda: Out of the twelve cases only one case (namely, Borda-DCPV-TE) was solved by Russel [140]. In particular, we solve all open cases of classical electoral control introduced by Bartholdi, Tovey, and Trick [10] and Hemaspaandra, Hemaspaandra, and Rothe [83] for Borda showing that Borda is resistant to all cases of constructive control and vulnerable to all but three cases of destructive control. We obtain our results for both winner models and also found two of the rare cases, namely destructive control by partition and by run-off partition of candidates with ties-promote tie-breaking, for which the complexity changes depending on which winner model is assumed.

Lastly, we study the model of online control, which was introduced by Hemaspaandra, Hemaspaandra, and Rothe [86, 87], showing that Borda is vulnerable against constructive and destructive online control by adding or deleting candidates and resistant against all types of online voter control (to be precise, we show coNP-hardness results for all cases).

3.2 Publication – Neveling and Rothe [120]

M. Neveling and J. Rothe. Control complexity in Borda elections: Solving all open cases of offline control and some cases of online control. *Artificial Intelligence*, 298:103508, 2021.

Preliminary versions of this paper were published in the proceedings of the *31st and the 33rd AAAI Conference on Artificial Intelligence* (AAAI'17 and AAAI'19, see [118, 137]) and of the *18th Italian Conference on Theoretical Computer Science* (ICTCS'17, see [117]).

3.3 Personal Contribution

The writing was done jointly with Jörg Rothe. All technical results are my contribution. Parts of this work already appeared in my Bachelor's and Master's Thesis. Specifically, Theorem 1 was part of my Bachelor's Thesis and Theorem 3, 10 and 13 were in my Master's Thesis.

CHAPTER 4

TOWARDS COMPLETING THE PUZZLE: COMPLEXITY OF CONTROL BY REPLACING, ADDING, AND DELETING CANDIDATES OR VOTERS

4.1 Summary

In this chapter we study various open problems regarding electoral control, therefore taking a step towards completing the puzzle of the complexity of electoral control problems for the most important voting rules. In particular, we initiate and complete the study of the standard control cases for plurality with runoff and veto with runoff.

We also study another special case of $\mathscr{E}$-CONSTRUCTIVE-MULTIMODE-CONTROL which models electoral control by replacing candidates or voters that was introduced by Loreggia et al. [102]. For replacement control the election chair may alter the set of candidates or set of voters while keeping the size of both sets the same as in the original election. For example, if a candidate is removed from the election, one candidate from the set of unregistered candidates must be added subsequently, thus replacing the removed candidate with the added candidate. To obtain the corresponding control problems we restrict $\mathscr{E}$-CONSTRUCTIVE-MULTIMODE-CONTROL as follows.

- For $\mathscr{E}$-CONSTRUCTIVE-CONTROL-BY-REPLACING-VOTERS we set $\ell_{AV} = \ell_{DV}$, $\ell_{AC} = \ell_{DC} = 0$, and $D = \emptyset$; and require in the question that $|V'| = |W'|$.
- For $\mathscr{E}$-CONSTRUCTIVE-CONTROL-BY-REPLACING-CANDIDATES we set $\ell_{AC} = \ell_{DC}$, $\ell_{AV} = \ell_{DV} = 0$, and $W = \emptyset$; and require in the question that $|C'| = |D'|$.

The complexity of replacement control problems are studied for Copeland$^{\alpha}$, maximin, k-veto, plurality/veto with runoff, Condorcet, fallback, and (normalized) range voting. We find that the complexity of replacement control always matches the complexity of the corresponding control by adding or deleting problem with the highest complexity. For example, plurality with runoff is in P for constructive control by adding candidates, constructive control by deleting candidates, and constructive control by replacing candidates. If for a voting rule the complexity of control by adding and control by deleting candidates or voters differs, then the complexity of control by replacing candidates or voters matches the higher complexity as is the case of maximin and constructive candidate control. It was shown by Loreggia et al. [102] that this is not necessarily the case. Interestingly, Condorcet and range voting are both immune against (constructive and destructive) control by adding candidates but in combination with control by deleting candidates they become susceptible to control.

4.2 Publication – Erdélyi, Neveling, Reger, Rothe, Yang and Zorn [58]

G. Erdélyi, M. Neveling, C. Reger, J. Rothe, Y. Yang, and R. Zorn. Towards completing the puzzle: Complexity of control by replacing, adding, and deleting candidates or voters. *Journal of Autonomous Agents and Multi-Agent Systems*. Submitted.

Preliminary versions of this paper were published in the proceedings of the *18th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS'19, see [60]) and of the *15th International Computer Science Symposium in Russia* (CSR'20, see [121]).

4.3 Personal Contribution

The writing was done jointly with my coauthors. Theorems 7, 8, and 20, and Theorems 22 – 25 are to be attributed to my contribution. Theorem 19 was done jointly with my coauthors.

CHAPTER 5

THE COMPLEXITY OF CLONING CANDIDATES IN MULTIWINNER ELECTIONS

5.1 Summary

We study how multiwinner elections can be tampered with by cloning candidates. For that we adapt the model for cloning candidates introduced by Elkind, Faliszewski, and Slinko [53] to multiwinner elections and define the following decision problems for a multiwinner voting rule $\mathscr{R}$.

$\mathscr{R}$-POSSIBLE-CLONING-GC	
Input:	A multiwinner election $E = (C,V,k)$, a cost function $\rho_i : \mathbb{N} \to \mathbb{N}$ for every $c_i \in C$, a distinguished candidate $p \in C$, and a budget B.
Question:	Is there a cloning vector $K = (K_1,\ldots,K_m)$ with $\sum_{c_i \in C} \rho_i(K_i) \leq B$ such that p (or one of her clones) is in a winning committee under $\mathscr{R}$ in at least one cloned multiwinner election E_K resulting from E via K?

$\mathscr{R}$-NECESSARY-CLONING-GC is defined analogously but we require that the cloning vector makes p (or one of her clones) part of a winning committee in *all* (instead of "at least one") cloned multiwinner elections resulting from E via the cloning vector.

Just like Elkind, Faliszewski, and Slinko [53] we study three cost models ZERO-COST, UNIT-COST, and GENERAL-COST. The corresponding problems with ZERO-COST and UNIT-COST are denoted by replacing "GC" with "ZC" and "UC", respectively, in the problem names. Our model is more focused than the singlewinner variant of Elkind, Faliszewski, and Slinko [53] in that we do not take on a probabilistic viewpoint in regards to when we view a cloning action as successful and only capture the extreme points by viewing a cloning action as successful if the distinguished candidate (or one of her clones) is in a winning committee in *at least one* or in *all* cloned multiwinner elections resulting from a cloning action.

We study the complexity of $\mathscr{R}$-POSSIBLE-CLONING-$\{$GC,UC,ZC$\}$ and $\mathscr{R}$-NECESSARY-CLONING-$\{$GC,UC,ZC$\}$ for the multiwinner voting rules defined in Chapter 2. In order to have polynomial-time winner determination for STV we use lexicographic tie-breaking for ties between candidates and arbitrary tie-breaking for ties between voters. Even with those simple tie-breaking rules both decision problems are intractable for STV: Possible cloning is NP-hard for STV in all cost models and coNP-hard for necessary cloning in all cost models. SNTV is easy for possible cloning and trivial for necessary cloning. Surprisingly, possible cloning with ZERO-COST and UNIT-COST is easy for k-Borda while it is NP-hard for Bloc. All other cases for Bloc and k-Borda are NP-hard. The reduction that is used to show NP-hardness of k-Borda-NECESSARY-CLONING-ZC also holds for $k = 1$ (i.e.,

the singlewinner variant of k-Borda) and therefore solves a problem left open by Elkind, Faliszewski, and Slinko [53].

The Chamberlin–Courant voting rules that we study have NP-hard winner determination meaning the decision problems defined above are trivially NP-hard for all cost models as well so we investigate the parameterized complexity of the problems with the number of candidates and the number of voters as parameters.

5.2 Publication – Neveling and Rothe [119]

M. Neveling and J. Rothe. The complexity of cloning candidates in multiwinner elections. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, pages 922–930. IFAAMAS, 2020.

5.3 Personal Contribution

The writing was done jointly with Jörg Rothe. Modeling and technical parts are to be attributed to my contribution.

CHAPTER 6

COMPLEXITY OF SHIFT BRIBERY FOR ITERATIVE VOTING RULES

6.1 Summary

We extend the study of shift bribery, introduced by Faliszewski et al. [66] and formally defined and studied by Elkind, Faliszewski, and Slinko [52], to iterative voting rules that elect the winner(s) of an election in multiple rounds.

For this chapter we will give an alternative but equivalent definition of shift bribery as it was defined in Chapter 2 that is more convenient to handle.

$\mathscr{E}$-CONSTRUCTIVE-SHIFT-BRIBERY	
Input:	An election (C,V) with n voters, a designated candidate $p \in C$, a budget B, and a list of price functions $\rho = (\rho_1, \ldots, \rho_n)$.
Question:	Is it possible to make p the unique $\mathscr{E}$-winner of the election by shifting p in the votes such that the total price does not exceed B?

For the destructive variant we are trying to prevent p from being the unique $\mathscr{E}$-winner. The price functions $\rho = (\rho_1, \ldots, \rho_n)$ with $\rho_i : \mathbb{N} \to \mathbb{N}$ describe how much it costs the briber to move p in a voter's vote forward (in the constructive case) or backward (in the destructive case). In particular, $\rho_i(k)$ is the cost of moving p in the ith voter's vote k positions forwards or backwards (for the constructive or destructive variant, respectively). To capture this behavior we require, for the constructive variant, that ρ_i is *nondecreasing* (i.e., $\rho_i(\ell) \leq \rho_i(\ell+1)$), $\rho_i(0) = 0$, and $\rho_i(\ell) = \rho_i(\ell-1)$ for each $\ell \geq r$ with r being the position of p in the (not-bribed) vote of voter i. Analogously, for the destructive variant we require that ρ_i is non*in*creasing, $\rho_i(0) = 0$, and $\rho_i(\ell) = \rho_i(\ell-1)$ for each $\ell \geq |C| - r + 1$ with r being the position of p in the (not-bribed) vote of voter i. For both variants the last condition is a technical requirement so that we cannot move p beyond the first or last position in a vote.

For all iterated variants of scoring rules defined in Chapter 2 we found that they are NP-hard for constructive and destructive shift bribery in both winner models. Furthermore, the price function as defined above only allows moving the designated candidate forwards in the constructive case (respectively, backwards in the destructive case) which makes sense for voting rules for which a candidate's final result in an election can only be improved if she is moved forward in the preferences of the voters. This so-called *monotonicity property* holds for scoring rules but does not hold for the iterative version except for iterated plurality and iterated veto. Therefore, we investigate whether the complexity changes if we drop the requirement that the designated candidate can only be shifted in one direction and give two examples of iterative scoring rules for which the problem still remains NP-hard.

We conjecture that the complexity also remains the same for all of our other nonmonotonic iterative scoring rules.

6.2 Publication – Maushagen, Neveling, Rothe, and Selker [106]

C. Maushagen, M. Neveling, J. Rothe, and A.-K. Selker. Complexity of shift bribery for iterative voting rules. *Journal of Autonomous Agents and Multi-Agent Systems*. Submitted.

A preliminary version of this paper was published in the proceedings of the *17th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS'18, see [107]).

6.3 Personal Contribution

The writing was done jointly with my co-authors. Theorems $2, 5, 6, 7, 8, 13$, and 14 are to be attributed to my contribution.

CHAPTER 7

CONCLUSIONS

We have studied how efficiently elections can be tampered with depending on which voting rule we choose to evaluate the election.

In Section 3 we have studied electoral control for the Borda Count and solved all open cases of standard electoral control and some cases of online electoral control. Borda turns out to be very resistant to constructive electoral control being resistant to all (standard) constructive types and in contrast vulnerable to most types of destructive electoral control. For future work we propose to solve the open problems of online candidate control, in particular involving partitioning the set of candidates. Furthermore, for the NP-hard cases parameterized complexity can be studied with, e.g., the number of candidates or voters as the parameter and for the vulnerable cases, the complexity of the more general cases with weighted elections can be studied. Lastly, the study of structured domains[1] (i.e., single-peaked and single-crossing elections) has been given attention lately [17, 22, 67, 159] and we propose to study whether the complexity of control for Borda changes if elections are structured.

In Section 4 we have studied electoral control focusing on control by replacing candidates or voters for various voting rules thus taking a step to complete the picture of complexity results regarding electoral control. One important case is still open which is constructive control by replacing candidates for 2-approval. The problem is seemingly related to the corresponding problem with 3-veto which is shown to be in P but the same approach cannot be used here. Since the problems for constructive control by *adding* candidates and by *deleting* candidates are in P for 2-approval, constructive control by replacing candidates is likely to be in P for 2-approval as well. On the contrary, showing that the problem is NP-hard would be interesting as we found that the complexity of replacement control usually follows the complexity of the corresponding problems of control by addition and deletion. Next, the problems for control by partitioning of candidates or voters are still open for plurality/veto with run-off. Lastly, showing dichotomy results for pure scoring rules similar to Hemaspaandra and Schnoor [90] is a challenging and interesting task.

In Section 5 we have devised a model for studying electoral control by cloning candidates in the setting of multiwinner elections and found a wide range of complexity results from easy cases like SNTV over cases that are easy in some ways but hard in others like k-Borda to cases like STV for which cloning is generally hard. We propose to solve the open cases regarding k-approval-CC and Borda-CC and extend our study to other multiwinner voting rules. In particular, Bredereck et al. [29] considered approximative versions of k-approval-CC and Borda-CC that only compute an approximated solution but run in polynomial time. Furthermore, it is interesting to study other classes of prize functions such

[1] Structured domains are motivated by the fact that, in practice, the voters' preferences are rarely purely random but structured in some way. For example, in political elections all candidates can be ordered on a left-right scale and voters tend to vote according to this scale. That is, a voter belonging to the left spectrum obviously prefers candidates on the left to candidates on the right.

as *all-or-nothing* prices. Lastly, our model could be extended to take on a probabilistic perspective similar to the model of cloning in singlewinner elections by Elkind, Faliszewski, and Slinko [53].

In Section 6 we have studied shift bribery for iterative scoring rule. We found that iterative scoring rules seem to be very resistant to shift bribery by showing NP-hardness of shift bribery for all iterative scoring rules that we have studied. In contrast, the standard non-iterative scoring rules are sometimes vulnerable to shift bribery as is the case for k-approval [52]. We have also investigated how nonmonotonicity affects the complexity of shift bribery by allowing the distinguished candidate to be shifted backwards in the constructive case and forwards in the destructive case and found no change in complexity. We propose to continue this study by proving or disproving our conjecture that using the nonmonotonicity of iterative scoring rules does not change the complexity of shift bribery for them. Since we have found exclusively NP-hardness results studying parameterized complexity for our problems with common parameters—the number of candidates, the number of voters, or the budget—seems natural. Recently, Zhou and Guo [160] started research in this direction by studying the parameterized complexity of shift bribery for four of our iterative voting rules finding a wide range of results including FPT and W[1]-hard cases. We propose to solve the cases that they left open and extend their study to other iterative voting rules. Moreover, we have studied iterative versions of the scoring rules plurality, veto, and Borda but there are many more scoring rules for which the iterative versions could be studied. It would be interesting to know if shift bribery is NP-hard for all of them which seems likely.

The study of the computational complexity of election tampering attempts (in particular, electoral control and bribery) has been a thriving research direction in computational social choice. In this thesis we have only covered worst-case complexity which admittedly is not the last word of wisdom. Rothe and Schend [139] argue that often times although some voting rule is resistant (i.e., NP-hard) against some form of election tampering on average the corresponding problem can be solved efficiently. Therefore, finding hardness in the average-case in addition to hardness in the worst-case is an interesting and important challenge for future work. Recently, Spielman, and Teng [148] proposed smoothed complexity theory which investigates the running time of algorithms when the input is randomly perturbed. In essence, smoothed complexity tries to answer the question of how robust or fragile worst-case instances of hard problems are. Therefore, smoothed complexity theory stands between worst-case and average-case analysis. Baumeister, Hogrebe, and Rothe [11] proposed to apply smoothed complexity to computational social choice. Furthermore, finding connections or interactions between the different subfields of computational social choice often yields interesting results as was done by Rey and Rothe [133] who, inspired by electoral control, have studied structural control in weighted voting games or Rothe, Schadrack, and Schend [138] who have used the Borda Count for FEN-hedonic games. Lastly, over the many years of research in computational social choice we have gained substantial insights into voting rules but applications of our insights besides for elections are sparse, so as a long term goal we propose to find new applications beyond computational social choice where our knowledge of voting rules becomes valuable.

BIBLIOGRAPHY

[1] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[2] K. Arrow. *Social Choice and Individual Values*. Wiley: New York, first edition, 1951.

[3] H. Aziz, M. Brill, v. Conitzer, E. Elkind, R. Freeman, and T. Walsh. Justified representation in approval-based committee voting. *Social Choice and Welfare*, 48(2):461–485, 2017.

[4] H. Aziz, E. Elkind, P. Faliszewski, M. Lackner, and P. Skowron. The Condorcet principle for multiwinner elections: From shortlisting to proportionality. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 84–90. IJCAI, 2017.

[5] H. Aziz, S. Gaspers, J. Gudmundsson, S. Mackenzie, N. Mattei, and T. Walsh. Computational aspects of multi-winner approval voting. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pages 107–115. IFAAMAS, 2015.

[6] H. Aziz, B. Lee, and N. Talmon. Proportionally representative participatory budgeting: Axioms and algorithms. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 23–31. IFAAMAS, July 2018.

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

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

[9] J. Bartholdi III, C. Tovey, and M. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(2):157–165, 1989.

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

[11] D. Baumeister, T. Hogrebe, and J. Rothe. Towards reality: Smoothed analysis in computational social choice. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, pages 1691–1695. IFAAMAS, May 2020.

[12] D. Baumeister and J. Rothe. Taking the final step to a full dichotomy of the possible winner problem in pure scoring rules. *Information Processing Letters*, 112(5):186–190, 2012.

[13] G. Benadè, S. Nath, A. Procaccia, and N. Shah. Preference elicitation for participatory budgeting. *Management Science*, 67(5):2813–2827, 2020.

[14] N. Betzler and B. Dorn. Towards a dichotomy for the possible winner problem in elections based on scoring rules. *Journal of Computer and System Sciences*, 76(8):812–836, 2010.

[15] N. Betzler, S. Hemmann, and R. Niedermeier. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 53–58. IJCAI, July 2009.

[16] N. Betzler, R. Niedermeier, and G. Woeginger. Unweighted coalitional manipulation under the Borda rule is NP-hard. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 55–60. AAAI Press/IJCAI, July 2011.

[17] N. Betzler, A. Slinko, and J. Uhlmann. On the computation of fully proportional representation. *Journal of Artificial Intelligence Research*, 47:475–519, 2013.

[18] D. Black. On the rationale of group decision-making. *Journal of Political Economy*, 56(1):23–34, 1948.

[19] D. Black. *The Theory of Committees and Elections*. Cambridge University Press, 1958.

[20] J.-C. de Borda. Mémoire sur les élections au scrutin. *Histoire de L'Académie Royale des Sciences, Paris*, 1781. English translation appears in the paper by de Grazia [42].

[21] S. Brams, P. Edelman, and P. Fishburn. Fair division of indivisible items. *Theory and Decision*, 55(2):147–180, 2003.

[22] F. Brandt, M. Brill, E. Hemaspaandra, and L. Hemaspaandra. Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. *Journal of Artificial Intelligence Research*, 53:439–496, July 2015.

[23] F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. Procaccia, editors. *Handbook of Computational Social Choice*. Cambridge University Press, 2016.

[24] R. Bredereck, J. Chen, P. Faliszewski, A. Nichterlein, and R. Niedermeier. Prices matter for the parameterized complexity of shift bribery. *Information and Computation*, 251:140–164, 2016.

[25] R. Bredereck, J. Chen, S. Hartung, S. Kratsch, R. Niedermeier, O. Suchý, and G. Woeginger. A multivariate complexity analysis of lobbying in multiple referenda. *Journal of Artificial Intelligence Research*, 50:409–446, 2014.

[26] R. Bredereck, J. Chen, R. Niedermeier, and T. Walsh. Parliamentary voting procedures: Agenda control, manipulation, and uncertainty. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 164–170. AAAI Press/IJCAI, July 2015.

[27] R. Bredereck, P. Faliszewski, A. Kaczmarczyk, R. Niedermeier, P. Skowron, and N. Talmon. Robustness among multiwinner voting rules. *Artificial Intelligence*, 290:Article 103403, 2021.

[28] R. Bredereck, P. Faliszewski, R. Niedermeier, and N. Talmon. Large-scale election campaigns: Combinatorial shift bribery. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pages 67–75. IFAAMAS, May 2015.

[29] R. Bredereck, P. Faliszewski, R. Niedermeier, and N. Talmon. Complexity of shift bribery in committee elections. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2452–2458. AAAI Press, February 2016.

[30] Y. Cabannes. Participatory budgeting: A significant contribution to participatory democracy. *Environment and Urbanization*, 16(1):27–46, 2004.

[31] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. In *Proceedings of the 33rd International Conference on Current Trends in Theory and Practice of Computer Science*, pages 51–69. Springer-Verlag *Lecture Notes in Computer Science #4362*, January 2007.

[32] A. Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58:345–363, 1936.

[33] J.-A.-N. de Caritat, Marquis de Condorcet. Plan de constitution, presenté à la convention nationale les 15 et 16 Fevrier 1793. In A. Condorcet O'Connor and M. Arago, editors, *Oeuvres de Condorcet, V. 12*. Firmin Didot Frères, 1847.

[34] V. Conitzer, M. Rognlie, and L. Xia. Preference functions that score rankings and maximum likelihood estimation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 109–115. IJCAI, July 2009.

[35] V. Conitzer and T. Sandholm. Vote elicitation: Complexity and strategy-proofness. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pages 392–397. AAAI Press, 2002.

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

[37] V. Conitzer and T. Walsh. Barriers to manipulation in voting. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 6, pages 127–145. Cambridge University Press, 2016.

[38] S. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing*, pages 151–158. ACM Press, 1971.

[39] M. Cygan, F. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.

[40] Baumeister D, S. Dennisen, and L. Rey. Winner determination and manipulation in minisum and minimax committee elections. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, pages 469–485. Springer-Verlag *Lecture Notes in Artificial Intelligence #9346*, September 2015.

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

[42] A. de Grazia. Mathematical deviation of an election system. *Isis*, 44(1–2):41–51, 1953.

[43] B. Debord. Prudent k-choice functions: Properties and algorithms. *Mathematical Social Sciences*, 26(1):63–77, 1993.

[44] C. Dodgson. Suggestions as to the best method of taking votes, where more than two issues are to be voted on. 1874. Reprints appear in [111, pp. 287–288] and in [19, pp. 222–224].

[45] B. Dorn and I. Schlotter. Multivariate complexity analysis of swap bribery. *Algorithmica*, 64(1):126–151, 2012.

[46] R. Downey and M. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.

[47] R. Downey and M. Fellows. *Fundamentals of Parameterized Complexity*, volume 4. Springer, 2013.

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

[49] B. Dutta, M. Jackson, and M. Le Breton. Strategic candidacy and voting procedures. *Econometrica*, 69(4):1013–1037, 2001.

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

[51] E. Elkind, P. Faliszewski, P. Skowron, and A. Slinko. Properties of multiwinner voting rules. *Social Choice and Welfare*, 48(3):599–632, 2017.

[52] E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. In *Proceedings of the 2nd International Symposium on Algorithmic Game Theory*, pages 299–310. Springer-Verlag *Lecture Notes in Computer Science #5814*, October 2009.

[53] E. Elkind, P. Faliszewski, and A. Slinko. Cloning in elections: Finding the possible winners. *Journal of Artificial Intelligence Research*, 42:529–573, 2011.

[54] E. Elkind and J. Rothe. Cooperative game theory. In J. Rothe, editor, *Economics and Computation. An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*, Springer Texts in Business and Economics, chapter 3, pages 135–193. Springer-Verlag, 2015.

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

[56] G. Erdélyi, M. Fellows, J. Rothe, and L. Schend. Control complexity in Bucklin and fallback voting: A theoretical analysis. *Journal of Computer and System Sciences*, 81(4):632–660, 2015.

[57] G. Erdélyi, E. Hemaspaandra, and L. Hemaspaandra. Bribery and voter control under voting-rule uncertainty. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems*, pages 61–68. IFAAMAS, May 2014.

[58] G. Erdélyi, M. Neveling, C. Reger, J. Rothe, Y. Yang, and R. Zorn. Towards completing the puzzle: Complexity of control by replacing, adding, and deleting candidates or voters. *Journal of Autonomous Agents and Multi-Agent Systems*. Submitted.

[59] G. Erdélyi, M. Nowak, and J. Rothe. Sincere-strategy preference-based approval voting fully resists constructive control and broadly resists destructive control. *Mathematical Logic Quarterly*, 55(4):425–443, 2009.

[60] G. Erdélyi, C. Reger, and Y. Yang. Towards completing the puzzle: Solving open problems for control in elections. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems*, pages 846–854. IFAAMAS, May 2019.

[61] R. Erikson. Malapportionment, gerrymandering, and party fortunes in congressional elections. *The American Political Science Review*, pages 1234–1245, 1972.

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

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

[64] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. Using complexity to protect elections. *Communications of the ACM*, 53(11):74–82, 2010.

[65] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. Multimode control attacks on elections. *Journal of Artificial Intelligence Research*, 40:305–351, 2011.

[66] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, 35:275–341, 2009.

[67] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. The shield that never was: Societies with single-peaked preferences are more open to manipulation and control. *Information and Computation*, 209(2):89–107, 2011.

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

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

[70] P. Faliszewski and J. Rothe. Control and bribery in voting. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. Procaccia, editors, *Handbook of Computational Social Choice*, chapter 7, pages 146–168. Cambridge University Press, 2016.

[71] P. Faliszewski, P. Skowron, A. Slinko, and N. Talmon. Multiwinner voting: A new challenge for social choice theory. In U. Endriss, editor, *Trends in Computational Social Choice*, chapter 2, pages 27–47. AI Access Foundation, 2017.

[72] P. Faliszewski, P. Skowron, A. Slinko, and N. Talmon. Multiwinner analogues of the plurality rule: Axiomatic and algorithmic perspectives. *Social Choice and Welfare*, 51(3):513–550, 2018.

[73] P. Faliszewski, P. Skowron, and N. Talmon. Bribery as a measure of candidate success: Complexity results for approval-based multiwinner rules. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, pages 6–14. IFAAMAS, May 2017.

[74] D. Felsenthal and Z. Maoz. Normative properties of four single-stage multi-winner electoral procedures. *Behavioral Science*, 37(2):109–127, 1992.

[75] P. Fishburn. *The Theory of Social Choice*. Princeton University Press, Princeton, N.J., 1973.

[76] J. Flum and M. Grohe. *Parameterized Complexity Theory*. EATCS Texts in Theoretical Computer Science. Springer-Verlag, 2006.

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

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

[79] M. Guo and V. Conitzer. Computationally feasible automated mechanism design: General approach and case studies. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 1676–1679. AAAI Press, July 2010.

[80] J. Hartmanis and R. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[81] T. Haynes, S. Sen, N. Arora, and R. Nadella. An automated meeting scheduling system that utilizes user preferences. In *Proceedings of the 1st International Conference on Autonomous Agents*, pages 308–315. ACM Press, 1997.

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

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

[84] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Hybrid elections broaden complexity-theoretic resistance to control. *Mathematical Logic Quarterly*, 55(4):397–424, 2009.

[85] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. The complexity of online manipulation of sequential elections. *Journal of Computer and System Sciences*, 80(4):697–710, 2014.

[86] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. The complexity of controlling candidate-sequential elections. *Theoretical Computer Science*, 678:14–21, 2017.

[87] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. The complexity of online voter control in sequential elections. *Journal of Autonomous Agents and Multi-Agent Systems*, 31(5):1055–1076, 2017.

[88] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. The complexity of online bribery in sequential elections. In *Proceedings of the 17th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 233–251. Electronic Proceedings in Theoretical Computer Science #297, July 2019.

[89] E. Hemaspaandra, L. Hemaspaandra, and H. Schnoor. A control dichotomy for pure scoring rules. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 712–720. AAAI Press, July 2014.

[90] E. Hemaspaandra and H. Schnoor. Dichotomy for pure scoring rules under manipulative electoral actions. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 1071–1079. IOS Press, August/September 2016.

[91] S. Issacharoff. Gerrymandering and political cartels. *Harvard Law Review*, pages 593–648, 2002.

[92] A. Kaczmarczyk and P. Faliszewski. Algorithms for destructive shift bribery. *Journal of Autonomous Agents and Multi-Agent Systems*, 33(3):275–297, 2019.

[93] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[94] J. Kemeny. Mathematics without numbers. *Dædalus*, 88(4):571–591, 1959.

[95] D. Kilgour. Approval balloting for multi-winner elections. In J. Laslier and R. Sanver, editors, *Handbook on Approval Voting*, chapter 6, pages 105–124. Springer, 2010.

[96] M. Kilgour, S. Brams, and R. Sanver. How to elect a representative committee using approval balloting. In F. Pukelsheim and B. Simeone, editors, *Mathematics and Democracy: Voting Systems and Collective Choice*, pages 83–95. Springer, 2006.

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

[98] K. Lam and C. Leung. Rank aggregation for meta-search engines. In *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters*, pages 384–385, 2004.

[99] J. Lang. Collective decision making under incomplete knowledge: Possible and necessary solutions. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 4885–4891. IJCAI, July 2020.

[100] J. Lang and J. Rothe. Fair division of indivisible goods. In J. Rothe, editor, *Economics and Computation. An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*, Springer Texts in Business and Economics, chapter 8, pages 493–550. Springer-Verlag, 2015.

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

[102] A. Loreggia, N. Narodytska, F. Rossi, B. Venable, and T. Walsh. Controlling elections by replacing candidates or votes (extended abstract). In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pages 1737–1738. IFAAMAS, May 2015.

[103] T. Lu and C. Boutilier. Budgeted social choice: From consensus to personalized decision making. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 280–286. AAAI Press/IJCAI, July 2011.

[104] T. Magrino, R. Rivest, E. Shen, and D. Wagner. Computing the margin of victory in IRV elections. In *Website Proceedings of the Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*, August 2011.

[105] J. Manza and C. Uggen. *Locked out: Felon disenfranchisement and American democracy*. Oxford University Press, 2008.

[106] C. Maushagen, M. Neveling, J. Rothe, and A.-K. Selker. Complexity of shift bribery for iterative voting rules. *Journal of Autonomous Agents and Multi-Agent Systems*. Submitted.

[107] C. Maushagen, M. Neveling, J. Rothe, and A.-K. Selker. Complexity of shift bribery in iterative elections. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 1567–1575. IFAAMAS, July 2018.

[108] C. Maushagen and J. Rothe. Complexity of control by partitioning veto elections and of control by adding candidates to plurality elections. *Annals of Mathematics and Artificial Intelligence*, 82(4):219–244, 2018.

[109] C. Maushagen and J. Rothe. The last voting rule is home: Complexity of control by partition of candidates or voters in maximin elections. In *Proceedings of the 24th European Conference on Artificial Intelligence*, pages 163–170. IOS Press, August/September 2020.

[110] K. May. A set of independent necessary and sufficient conditions for simple majority decision. *Econometrica: Journal of the Econometric Society*, pages 680–684, 1952.

[111] I. McLean and A. Urken. *Classics of Social Choice*. University of Michigan Press, 1995.

[112] R. Meir, M. Polukarov, J. Rosenschein, and N. Jennings. Convergence to equilibria in plurality voting. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 823–828. AAAI Press, 2010.

[113] R. Meir, A. Procaccia, J. Rosenschein, and A. Zohar. Complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research*, 33:149–178, 2008.

[114] C. Menton. Normalized range voting broadly resists control. *Theory of Computing Systems*, 53(4):507–531, 2013.

[115] A. Meyer and L. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 125–129. IEEE Computer Society Press, 1972.

[116] Nina Narodytska and Toby Walsh. Manipulating two stage voting rules. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pages 423–430. IFAAMAS, May 2013.

[117] M. Neveling and J. Rothe. Closing the gap of control complexity in Borda elections: Solving ten open cases. In *Proceedings of the 18th Italian Conference on Theoretical Computer Science*, volume 1949, pages 138–149. CEUR-WS.org, September 2017.

[118] M. Neveling and J. Rothe. Solving seven open problems of offline and online control in Borda elections. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 3029–3035. AAAI Press, February 2017.

[119] M. Neveling and J. Rothe. The complexity of cloning candidates in multiwinner elections. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, pages 922–930. IFAAMAS, May 2020.

[120] M. Neveling and J. Rothe. Control complexity in Borda elections: Solving all open cases of offline control and some cases of online control. *Artificial Intelligence*, 298:Article 103508, 2021.

[121] M. Neveling, J. Rothe, and R. Zorn. The complexity of controlling Condorcet, fallback, and k-veto elections by replacing candidates or voters. In *Proceedings of the 15th International Computer Science Symposium in Russia*, pages 314–327. Springer, June 2020.

[122] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[123] S. Obraztsova, Y. Zick, and E. Elkind. On manipulation in multi-winner elections based on scoring rules. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pages 359–366. IFAAMAS, 2013.

[124] K. Oflazer and G. Tür. Morphological disambiguation by voting constraints. In *Proceedings of the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 222–229. ACL/Morgan Kaufmann, 1997.

[125] Y. Narahari P. Dey, N. Misra. Frugal bribery in voting. *Theoretical Computer Science*, 676:15–32, 2017.

[126] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, second edition, 1995.

[127] D. Parkes and L. Xia. A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 1429–1435. AAAI Press, July 2012.

[128] D. Pennock, E. Horvitz, and C. Giles. Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 729–734. AAAI Press, July/August 2000.

[129] D. Peters. Proportionality and strategyproofness in multiwinner elections. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, page 1549–1557. IFAAMAS, July 2018.

[130] M. Pini, F. Rossi, B. Venable, and T. Walsh. Dealing with incomplete agents' preferences and an uncertain agenda in group decision making via sequential majority voting. In *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning*, pages 571–578. AAAI Press, September 2008.

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

[132] A. Procaccia, J. Rosenschein, and A. Zohar. On the complexity of achieving proportional representation. *Social Choice and Welfare*, 30(3):353–362, 2008.

[133] A. Rey and J. Rothe. Structural control in weighted voting games. *The B.E. Journal on Theoretical Economics*, 18(2), 2018.

[134] A. Roth, T. Sönmez, and M. Ünver. Kidney exchange. *The Quarterly Journal of Economics*, 119(2):457–488, 2004.

[135] J. Rothe. *Complexity Theory and Cryptology. An Introduction to Cryptocomplexity*. EATCS Texts in Theoretical Computer Science. Springer-Verlag, 2005.

[136] J. Rothe, editor. *Economics and Computation. An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*. Springer Texts in Business and Economics. Springer-Verlag, 2015.

[137] J. Rothe. Borda count in collective decision making: A summary of recent results. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 9830–9836. AAAI Press, January/February 2019.

[138] J. Rothe, H. Schadrack, and L. Schend. Borda-induced hedonic games with friends, enemies, and neutral players. *Mathematical Social Sciences*, 96:21–36, 2018.

[139] J. Rothe and L. Schend. Challenges to complexity shields that are supposed to protect elections against manipulation and control: A survey. *Annals of Mathematics and Artificial Intelligence*, 68(1–3):161–193, 2013.

[140] D. Russo. *Structural Properties of Complexity Classes*. PhD thesis, University of California at Santa Barbara, Santa Barbara, CA, 1985.

[141] D. Saari. Which is better: The Condorcet or Borda winner? *Social Choice and Welfare*, 27(1):107–129, 2006.

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

[143] I. Schlotter, P. Faliszewski, and E. Elkind. Campaign management under approval-driven voting rules. *Algorithmica*, 77:84–115, 2017.

[144] S. Sekar, S. Sikdar, and L. Xia. Condorcet consistent bundling with social choice. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, pages 33–41. IFAAMAS, May 2017.

[145] G. Sigletos, G. Paliouras, C. Spyropoulos, and M. Hatzopoulos. Combining information extraction systems using voting and stacked generalization. *Journal of Machine Learning Research*, 6(11):1751–1782, 2005.

[146] P. Skowron, P. Faliszewski, and J. Lang. Finding a collective set of items: From proportional multirepresentation to group recommendation. *Artificial Intelligence*, 241:191–216, 2016.

[147] P. Skowron, P. Faliszewski, and A. Slinko. Axiomatic characterization of committee scoring rules. *Journal of Economic Theory*, 180:244–273, 2019.

[148] D. Spielman and Teng S. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

[149] L. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[150] N. Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206, 1987.

[151] C. Tovey. Tutorial on computational complexity. *Interfaces*, 32(3):30–61, 2002.

[152] A. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, ser. 2*, 42:230–265, 1936. Correction, *ibid*, vol. 43, pp. 544–546, 1937.

[153] T. Walsh. Uncertainty in preference elicitation and aggregation. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 3–8. AAAI Press, July 2007.

[154] M. Weston. One person, no vote: Staggered elections, redistributing and disenfranchisement. *Yale LJ*, 121:2013, 2011.

[155] L. Xia. Computing the margin of victory for various voting rules. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 982–999. ACM Press, June 2012.

[156] L. Xia and V. Conitzer. Determining possible and necessary winners given partial orders. *Journal of Artificial Intelligence Research*, 41:25–67, 2011.

[157] L. Xia, V. Conitzer, and A. Procaccia. A scheduling approach to coalitional manipulation. In *Proceedings of the 11th ACM Conference on Electronic Commerce*, pages 275–284. ACM Press, June 2010.

[158] L. Xia, M. Zuckerman, A. Procaccia, V. Conitzer, and J. Rosenschein. Complexity of unweighted coalitional manipulation under some common voting rules. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 348–353. IJCAI, July 2009.

[159] Y. Yang. On the complexity of Borda control in single-peaked elections. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, pages 1178–1186. IFAAMAS, May 2017.

[160] A. Zhou and J. Guo. Parameterized complexity of shift bribery in iterative elections. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, pages 1665–1673. IFAAMAS, May 2020.

Eidesstattliche Erklärung

entsprechend §5 der Promotionsordnung vom 15.06.2018.

Ich versichere an Eides Statt, dass die Dissertation von mir selbständig und ohne unzulässige fremde Hilfe unter Beachtung der „Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf“ erstellt worden ist. Desweiteren erkläre ich, dass ich eine Dissertation in der vorliegenden oder in ähnlicher Form noch bei keiner anderen Institution eingereicht habe.

Teile dieser Dissertation wurden bereits in Form folgender Zeitschriftenartikel und Konferenzberichte veröffentlicht oder zur Begutachtung eingereicht und sind entsprechend gekennzeichnet: [58], [60], [107], [106], [120], [117], [118], [121], [119], [137].

Ort, Datum	Marc Neveling