Studies in Computational Aspects of Voting
— a Parameterized Complexity Perspective*

Dedicated to Michael R. Fellows on the occasion of his 60th birthday

Nadja Betzler, Robert Bredereck, Jiehua Chen, and Rolf Niedermeier

Institut für Softwaretechnik und Theoretische Informatik,
TU Berlin, Germany
{robert.bredereck, jiehua.chen, rolf.niedermeier}@tu-berlin.de

Abstract. We review NP-hard voting problems together with their status in terms of parameterized complexity results. In addition, we survey standard techniques for achieving fixed-parameter (in)tractability results in voting.

1 Introduction

Once there is more than one alternative for a community to choose from, voting comes into play. Different voters usually have conflicting preferences over the alternatives, hence some voting protocol has to be used to reach a joint decision or, in other words, to aggregate preferences. Voting is part of the fields of preference handling, decision making, and social choice. There are many voting protocols whose pros and cons have been studied for centuries in such diverse fields as philosophy, mathematics, political science, and economy. Recently, computer science has entered the stage for several reasons. With the omnipresence of the Internet and modern communication tools, applications such as auctions, bids, ratings, and rankings have become an everyday business. All these are related to voting scenarios. Moreover, the advent of intelligent multi-agent systems leads to numerous cases of preference aggregation. Inside computer science, voting occurs in quite diverse areas, including planning problems in multi-agent systems [ER91,ER97], spam detection [DKNS01a], databases [FKS03], bioinformatics [JSA08], and graph drawing [BBD09]. We refer interested readers to a couple of surveys [BCE12,BEH+10,CELM07,Con10,FHH10,FHHR09a] and a book [RBLR11, in German] for a general overview on voting in computer science.

Voting problems (winner determination being just the most basic one) come in many different guises, often making the corresponding tasks computationally challenging to solve. First of all, there are numerous different voting protocols including Plurality, k-Approval, and Kemeny, to name just a few. Then, it may happen that there are only incomplete voter preferences available, making the determination of a possible or necessary winner hard. Moreover, questions such as manipulation, control, or bribery often lead to NP-hard problems. The study

* Supported by the DFG, research project PAWS, NI 369/10.

of the computational complexity of voting problems was initiated by a seminal series of papers of Bartholdi, Orlin, Tovey, and Trick [BO91,BTT89a,BTT89b]. Many voting problems turned out to be NP-hard. Actually, Bartholdi et al. pointed out that in the context of voting, computational intractability may sometimes be a desirable property. For instance, it is desirable to have a voting protocol that is "resistant" against attacks such as manipulation or bribery.

Voting problems carry many natural parameters, obviously including the number of candidates and the number of votes. There are real-world scenarios for each of them having small values. Hence, the analysis of parameterized computational complexity comes into play. To the best of our knowledge, this fruitful line of research was *explicitly* initiated Christian, Fellows, Rosamond, and Slinko [CFRS07] in a work concerned with lobbying. Moreover, a complexity analysis for manipulating voting systems when the parameter "number of candidates" is small was addressed by Conitzer, Sandholm, and Lang [CSL07]. In this survey, we try to review the state of the art and motivate the rapidly developing field of parameterized complexity analysis for voting problems. See Lindner and Rothe [LR08] for an early survey in this direction.

Our work is organized as follows. Section 2 introduces some basic concepts and definitions related to both voting problems and parameterized complexity analysis. In Section 3, we briefly review a number of prominent voting protocols and some of their respective pros and cons. In Section 4, we survey in some detail the state of the art concerning the multivariate complexity analysis for Kemeny voting. This exhibits how many different parameters naturally occur in a practically relevant voting problem, and how the tools of parameterized complexity analysis can help to better understand the computational complexity of an NP-hard voting problem. In Section 5, we present several NP-hard voting problems and describe their status in terms of parameterized complexity analysis. In Section 6, we describe applications of tools from parameterized algorithmics that have been applied to gain fixed-parameter tractability results for voting problems. Finally, in Section 7, we discuss the relevance and benefits of parameterized (and multivariate) complexity analysis in voting scenarios and conclude with numerous challenges for future research.

2 Preliminaries

Since we are talking about voting problems and their computational complexity, we start with basic definitions from the context of voting. We assume familiarity with *classical computational complexity theory* [Pap94,AB09], and we provide some basic definitions concerning *parameterized computational complexity theory* [DF99,FG06,Nie06].

Formally, an *election* (C, V) consists of a set C of m *candidates* (or, synonymously, alternatives) and a multiset V of n *votes*. If not stated otherwise, a *vote* is a linear order (that is, a transitive, antisymmetric, and total relation) on C. Sometimes we also call this a *ranking* over C. For example, for $C = \{a, b, c\}$, the vote $a \succ_v b \succ_v c$ expresses that a is the best-liked and c the least-liked candidate

in the vote v. We use '$\succ$' instead of '$\succ_v$' if it is clear from the context which vote we mean. For any two candidates $a \neq b$, let $\#(a, b)$ be the number of votes that rank candidate a higher than candidate b in the considered election. A *voting protocol*[1] is a function that maps an election to a subset of candidates, the set of winners. When one is interested in finding a uniquely determined winner (that is, a one-element winner set), one refers to such a candidate as *unique winner*. When allowing for a set of winners, the corresponding candidates are denoted as *co-winners*.

Sometimes we also consider a more general definition of votes. There are scenarios where (complete) linear orders are not available. That is, some candidates are not comparable in some votes, leading to *incomplete votes*. In such cases, our votes are partial orders on the candidate set. A linear order v *extends* a partial order w if $w \subseteq v$, that is, for any $c_1, c_2 \in C$ one has $c_1 \succ_w c_2 \Rightarrow c_1 \succ_v c_2$. Consider two candidates $a, b \in C$ and an incomplete vote $v \in V$. If neither $a \succ_v b$ nor $b \succ_v a$, then we say the candidate pair $\{a, b\}$ is *undetermined* in vote v.

Parameterized Complexity. The concept of parameterized complexity was pioneered by Downey and Fellows [DF99] (see also [FG06,Nie06] for more recent textbooks). The fundamental goal is to find out whether the seemingly unavoidable combinatorial explosion occurring in algorithms to decide NP-hard problems can be confined to certain problem-specific parameters. The idea is the following: When such a parameter assumes only small values in applications, then an algorithm with a running time that is exponential exclusively with respect to the parameter may be efficient and practical. We now provide some formal definitions.

Definition 1 (Parameterized Problem). *A* parameterized problem *is a language $L \subseteq \Sigma^* \times \Sigma^*$, where Σ is an alphabet. The second component is called the* parameter *of the problem.*

We typically consider the special case of parameters which are non-negative integers or "combined" parameters which are tuples of non-negative integers. For instance, an obvious parameter in voting is the number of candidates. Thus, typically $L \subseteq \Sigma^* \times \mathbb{N}$, where a combined parameter can be interpreted as the maximum of its integer components.

Definition 2 (Fixed-Parameter Tractability). *A parameterized problem L is* fixed-parameter tractable *if there is an algorithm that decides in $f(k) \cdot |x|^{\mathcal{O}(1)}$ time whether $(x, k) \in L$, where f is an arbitrary computable function depending only on k. Correspondingly,* FPT *denotes the class of all fixed-parameter tractable parameterized problems.*

[1] In this survey, we do not discuss the more general concepts of *social choice functions* or *social welfare functions*. Note that by our definition of voting protocols, every voting protocol is *anonymous*, that is, the voting protocol does not discriminate among voters. We will only exemplarily discuss some other properties of the considered voting protocols when necessary. For an overview about general concepts and properties of voting protocols, we refer to the two handbooks on social choice and welfare [ASS02,ASS10].

We stress that the concept of fixed-parameter tractability is different from the notion of "polynomial-time solvability for constant k" since an algorithm running in $\mathcal{O}(|x|^k)$ time does not show fixed-parameter tractability. All problems which can be solved in running time $|x|^{f(k)}$ for a computable function f form the complexity class XP, where $f : \mathbb{N} \to \mathbb{N}$ is a function depending only on k. Clearly, FPT $\subseteq$ XP.

For many parameterized problems, fixed-parameter tractability could not be shown. Downey and Fellows [DF99] developed a theory of (presumable) *parameterized intractability*. It comprises of a hierarchy of complexity classes coming along with complete problems. This so-called *W-hierarchy* consists of the following classes and interrelations:

$$\text{FPT} \subseteq \text{W[1]} \subseteq \text{W[2]} \subseteq \cdots \subseteq \text{W[Sat]} \subseteq \text{W[P]} \subseteq \text{XP}.$$

In this survey, we only provide intractability results regarding the first two levels of (presumable) parameterized intractability captured by the complexity classes W[1] and W[2]. The containment W[1] $\subseteq$ FPT would not imply P = NP but the failure of the Exponential Time Hypothesis [IP01,IPZ01].[2] It is commonly believed that W[1]-hard problems are not fixed-parameter tractable. To show W[t]-hardness for any non-negative integer t, we introduce the following reducibility concept.

Definition 3 (Parameterized Reduction). *Let $L, L' \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. A* parameterized reduction *from L to L' consists of two mappings $\phi : \Sigma^* \times \mathbb{N} \to \Sigma^*$ and $g : \mathbb{N} \to \mathbb{N}$, where for every $x \in \Sigma^*$ and $k \in \mathbb{N}$ it holds that*

- *$(x, k) \mapsto \phi(x, k)$ is computable in time $h(k) \cdot |x|^{\mathcal{O}(1)}$ with $h : \mathbb{N} \to \mathbb{N}$, and*
- *$(x, k) \in L \Leftrightarrow (\phi(x, k), g(k)) \in L'$.*

Analogously to the case of NP-hardness, for any non-negative integer t, it suffices to give a parameterized reduction from one W[t]-hard parameterized problem X to a parameterized problem Y to show W[t]-hardness of Y. Containment of Y in W[t] can be shown by giving a reduction from Y to a problem contained in W[t]. If there are parameterized reductions for two problems such that each of them can be reduced to the other problem, we say that they are *FPT-equivalent*.

Kernelization [Bod09,GN07] is an alternative way of showing fixed-parameter tractability [CCDF97]. In a nutshell, it is a polynomial-time algorithm that transforms an instance of a parameterized problem into an equivalent instance whose size is bounded by a function of the parameter. This resulting instance is called a *problem kernel*. Typically, kernelizations are based on several polynomial-time executable *data-reduction rules* that help shrinking the instance size.

For more details about parameterized complexity theory we refer to the textbooks [DF99,FG06,Nie06] and a recent survey by Downey and Thilikos [DT11].

[2] In a nutshell, the Exponential Time Hypothesis says that, for $k \geq 3$, the NP-complete k-SAT problem cannot be solved in time subexponential in the number of variables.

Table 1. Hypothetical student rankings of TU Berlin (B), MIT (M), Oxford University (O), Tsinghua University (T) and ETH Zurich (Z) according to research in parameterized complexity, salary, practicing English, and cultural activities, respectively.

Criterion	Institutions
Parameterized complexity	$B \succ O \succ M \succ T \succ Z$
Salary	$Z \succ O \succ M \succ T \succ B$
Practicing English	$M \succ O \succ B \succ Z \succ T$
Cultural activities	$B \succ T \succ Z \succ M \succ O$

3 Types of Voting Protocols

Suppose that a student decides to pursue his PhD in parameterized complexity analysis. He gets five offers: From TU Berlin (B), MIT (M), Oxford University (O), Tsinghua University (T), and ETH Zurich (Z). He decides to select one that is not only good in research but also offers a manifold of cultural activities, as well as a good opportunity to polish his English. Last but not least, he needs a decent income.

Depending on these different criteria, the five universities are ranked[3]. In the field of parameterized complexity, TU Berlin is ranked first, followed by Oxford University, MIT, and Tsinghua University. ETH Zurich is ranked last. As for salaries, ETH Zurich makes the best offer, followed by Oxford University, MIT, and Tsinghua University. TU Berlin offers the least. For practicing English, MIT ranks first, followed by Oxford University, and then by TU Berlin. ETH Zurich is ranked fourth and Tsinghua University last. With respect to cultural activities, TU Berlin is ranked the best, followed by Tsinghua University, and then by ETH Zurich. Oxford University is ranked last, just behind MIT. These rankings are listed in Table 1.

The universities B, M, O, T, and Z can be seen as the candidates for an election and the rankings in Table 1 as the votes on these candidates. Deciding on an institution means aggregating the different rankings and deciding on the winner of this election.

Applying different voting protocols to the same multiset of votes may lead to different winners. Many of the most widely used voting protocols can be assigned to one of the following two classes: *scoring protocols* and *voting protocols based on pairwise comparisons between candidates*. In the following, we will look into these two classes in some detail (Sections 3.1 and 3.2). In Section 3.3, we will take a look at some additional voting protocols which fall into neither class. We illustrate some common and popular voting protocols with the help of our PhD place example. We emphasize that it would be beyond the scope of our survey to name

[3] Clearly, these rankings are influenced by marketing and political pressure. In this example, also a certain degree of bribery comes into play.

and discuss the properties (desirable and undesirable ones) of the various voting protocols. For further information on this topic, or voting protocols in general, we refer to the expositions of Arrow et al. [ASS02,ASS10], Gaertner [Gae09], Nurmi [Nur87], Rothe et al. [RBLR11, in German], and Taylor [Tay05].

3.1 Scoring Protocols

In a (positional) scoring protocol, each candidate is assigned a certain number of points from each vote depending on her position in this vote. A candidate is called a winner if no other candidate gets a higher total sum of points.

Plurality. Plurality is perhaps the most widely used voting protocol. It is a scoring protocol which assigns one point to the top-ranked candidate in each vote, and zero points to all others. A candidate with the highest total score belongs to the winner set. There may be multiple winners. In our PhD place example, it is easily seen that the winning university is TU Berlin (2 points) if we use the Plurality protocol to make the decision.

Plurality is very simple. However, it has some shortcomings. For example, it is often criticized for considering only the topmost candidate of each vote and completely disregarding the information about other candidates. For this reason voters sometimes do not submit their true preferences if they know that their most preferred candidate has only a small chance to win. Suppose that there is an election on three candidates, a, b, and c, with

two votes $a \succ b \succ c$,
four votes $c \succ b \succ a$, and
three vote $b \succ a \succ c$.

According to the Plurality voting protocol, candidate c wins with four points. However, if the first two voters exchange the positions of candidates a and b in their votes such that they submit $b \succ a \succ c$ instead of $a \succ b \succ c$, then candidate b wins with five points. This is a better outcome for them, since they prefer candidate b to candidate c.

k-Approval. Occasionally, a voter has more than one favorite candidate. The k-Approval voting protocol gives the possibility to "approve" k candidates: The first k candidates in a vote get one point each. Thus, Plurality is the same as 1-Approval. In our example, using 2-Approval, one would select Oxford University (3 points) for a PhD position, which intuitively seems to be a good compromise, since three of four criteria are ranking Oxford University in the second position.

Veto. Another simple scoring protocol is Veto. It assigns zero points to the last candidate and one point to each of the other candidates in each vote. Once again, every candidate with the highest sum of points wins. Using Veto, the PhD place example will result in selecting MIT (4 points). Veto is the same as $(m-1)$-Approval, where m is the total number of candidates.

Borda. A prominent voting protocol is Borda voting.[4] Borda voting directly translates the position of each candidate in a vote into the number of points she gets. For each vote, Borda voting assigns zero points to the candidate ranked last, one point to the candidate ranked last but one, etc., and the highest-ranked candidate in each vote is assigned $m-1$ points. Once again, every candidate with the highest total score wins. According to Borda voting, TU Berlin (10 points) is the winner in the PhD place example.

Determining the set of winners using any of the scoring protocols described above can be easily done in time polynomial in the input size.

3.2 Voting Protocols Based on Pairwise Comparisons

Comparison-based voting protocols date back to the 13th century. Ramon Lull, who first came up with Borda voting, devised a voting protocol which takes into account pairwise comparisons between any two candidates. Today, this is known as the *Condorcet method* [dC85].[5]

Definition 4 (Condorcet winner). *A candidate is the* Condorcet winner *if she is preferred to any other candidate in more than half of the votes.*

Obviously, deciding whether a candidate is the Condorcet winner can be done efficiently, that is, in polynomial time. However, not every election has a Condorcet winner. For instance, in the PhD place example, there is no Condorcet winner since no institution has an absolute majority of votes which prefer it to any other institution: TU Berlin beats both, Tsinghua University and ETH Zurich by 3-to-1; TU Berlin and Oxford University, TU Berlin and MIT as well as Oxford University and MIT are tied 2-to-2. The pairwise comparisons of every two candidates are shown in Table 2.

There is a close relation between directed graphs and voting protocols based on pairwise comparisons. More precisely, for each election there is a *majority graph* which is defined as follows:

Definition 5 (Majority Graph). *The* majority graph *of an election $E = (C, V)$ is a directed graph whose vertices are the candidates and there is an arc from vertex v to vertex w if and only if more than half of the votes prefer candidate v to candidate w. An arc from v to w is labeled with "$x : y$" which means that x votes prefer v to w, and y votes prefer w to v.*

[4] The Borda voting protocol was invented independently several times. It was first described by Ramon Llull, a 13th century Majorcan writer and philosopher. It is now named after Jean-Charles de Borda, a French mathematician, physicist, political scientist of the 18th century [dB81].

[5] Named after the 18th-century French philosopher Marie Jean Antonie Nicolas de Caritat, Marquis de Condorcet.

Table 2. Pairwise comparisons in the PhD place example

Candidate pairs (x, y)	# votes with $x \succ y$	# votes with $y \succ x$
(B, O)	2	2
(B, M)	2	2
(B, T)	3	1
(B, Z)	3	1
(O, M)	2	2
(O, T)	3	1
(O, Z)	2	2
(M, T)	3	1
(M, Z)	2	2
(T, Z)	2	2

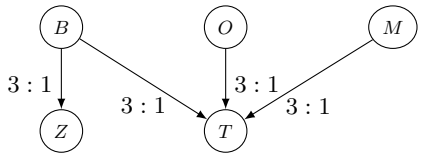

Fig. 1. The majority graph of our PhD place example.

Many voting problems (especially when comparison-based voting protocols are involved) can be considered as directed (weighted) graph problems. For example, if there is a vertex with exactly $m - 1$ outgoing arcs with m being the number of candidates, then the corresponding candidate is the Condorcet winner. As we can easily see from the majority graph of our example in Figure 1, no vertex with out-degree four exists. This meets with the fact that there is no Condorcet winner in our PhD place example.

Although the existence of a Condorcet winner cannot be guaranteed, the Condorcet winner for an election is always unique if it does exist. Many voting protocols are designed to choose a candidate as the winner who is "closest" to the Condorcet winner. In the following, we will take a closer look at five well-known comparison-based voting protocols (Dodgson, Kemeny, Young, Copeland$^\alpha$, and Maximin [Dod76,Kem59,You77,Cop51,Wal49,Fis77]) which all fulfill the *Condorcet principle*, that is, the Condorcet winner for an election will be selected as the winner if she exists. The winner determination problems for the first three voting protocols are NP-hard, while the last two can be solved efficiently, that is, in time polynomial in the input size.

Dodgson Voting. In his work "A Method of Taking Votes on More than Two Issues" [Dod76], the English writer, mathematician, and logician Charles Lutwidge

Dodgson (better known as Lewis Carroll) proposed selecting the winner set as follows: Any candidate requiring the minimum number of *swaps* between two neighboring candidates to become a Condorcet winner is considered as a winner. Given an election and a non-negative integer $k \in \mathbb{N}$, determining whether a candidate can become a Condorcet winner with at most k swaps in the given votes is NP-complete [BTT89b]. This problem is called Dodgson Score. In the PhD place example, the Dodgson score of TU Berlin is 2: By exchanging the positions of TU Berlin and Oxford University, and then the positions of TU Berlin and MIT, the ranking with respect to "Practicing English" turns into $\mathrm{B} \succ \mathrm{M} \succ \mathrm{O} \succ \mathrm{Z} \succ \mathrm{T}$ and TU Berlin becomes the Condorcet winner. In fact, this is the fewest number of swaps needed to let the PhD place example have a Condorcet winner. Finally, we remark that generalized winner determination for Dodgson is complete for parallel access to NP ($\mathbf{P}_{||}^{\mathbf{NP}}$-complete) [HHR97].

Kemeny Voting. This voting protocol goes back to Kemeny [Kem59] and Condorcet [dC85] and was specified by Levenglick [Lev75] (see also our case study for Kemeny voting in Section 4). Consider an election consisting of a multiset of rankings of the candidates. The *Kendall-Tau distance* between rankings r_1 and r_2 is the number of swaps of two neighboring candidates in order to transform r_1 into r_2. The *score* of a ranking r is the sum of Kendall-Tau distances between r and each input ranking. A "consensus ranking" with respect to Kemeny voting is a ranking with minimum score. Correspondingly, we call such a ranking a *Kemeny consensus*. The first candidate in a Kemeny consensus is considered as a winner. TU Berlin as well as Oxford University are winners in our PhD place example. See Section 4 for more details.

Kemeny voting has many desirable properties. For example, it is the only voting protocol which is *neutral* and *consistent*[6], and satisfies the Condorcet principle [YL78]. Thus, Kemeny voting is used in various applications such as meta-search engines, spam detection [DKNS01a], databases [FKS03,Scu07], or the construction of genetic maps in bioinformatics [JSA08]. However, to determine a Kemeny consensus is computationally intractable. More precisely, the Kemeny Score problem, that is, given an election and a non-negative integer $k \in \mathbb{N}$, determining whether the score of a Kemeny consensus is at most k is NP-complete [BTT89b]. Some more general Kemeny voting related problems (including winner determination) are even $\mathbf{P}_{||}^{\mathbf{NP}}$-complete [HSV05].

Young Voting. H. Peyton Young [You77] took a different approach to finding a candidate "closest" to the Condorcet winner. His main idea was to delete the fewest number of votes to let the remaining votes have a Condorcet winner. The Young Score problem asks whether a candidate can become the Condorcet

[6] A voting protocol is *neutral* if the candidates are treated equally, that is, if the candidates of an election are renamed, the winner of the election with renamed candidates is the renamed winner of the original election. *Consistency* requires that if a candidate wins in two multisets of votes, then she should also win in the union multiset of these two multisets.

winner in a "sub-election" consisting of at least k' ($k' \in \mathbb{N}$) of the input votes, while the DUAL YOUNG SCORE problem asks whether deleting at most k ($k \in \mathbb{N}$) votes can make a distinguished candidate the Condorcet winner. Both problems are NP-complete [RSV03]. For the PhD place example, removing only one vote (the ranking for practicing English) can make the remaining votes have a Condorcet winner (TU Berlin). Finally, we remark that the YOUNG WINNER problem, that is, deciding whether a distinguished candidate can become a Condorcet winner by the deleting minimum number of votes, is $\mathbf{P}_{||}^{\mathbf{NP}}$-complete [RSV03]. As one can easily verify, in our PhD place example, TU Berlin is a Young winner.

The NP-hardness results for the winner determination problems described above make such voting protocols usually infeasible for practical use. However, in some restricted scenarios winner determination becomes efficiently solvable. For example, DODGSON SCORE is fixed-parameter tractable with respect to the number of candidates. Table 4 in Section 5.1 gives some parameterized complexity results for DODGSON SCORE, DUAL YOUNG SCORE, and YOUNG SCORE, while the corresponding analysis of Kemeny voting is discussed in more detail in our case study (Section 4).

Copeland$^\alpha$ Voting. This voting protocol considers each pair of candidates: The candidate that beats the other one in more than half of the votes is rewarded one point, while the loser gets zero points. If the two candidates are tied, then each gets α points. The candidate with highest total score wins. The original Copeland voting [Cop51,BF02,Goo54] uses a slightly different way for awarding points to the loser in a pairwise comparison. However, it is equivalent to Copeland$^{0.5}$ [FHHR09b]. In our PhD place example, TU Berlin wins with $(2 + 2 \cdot \alpha)$ points under Copeland$^\alpha$ voting (see Figure 1). For $\alpha = 1$, there are also two more co-winners (Oxford University and MIT).

Maximin Voting. Let $\#_{\min}(x) = \min\{\#(x, y) : y \in C \setminus \{x\}\}$ for $x \in C$ (recall that $\#(c, c')$ is the number of votes ranking candidate c higher than candidate c'). According to Maximin voting, a candidate c wins if she has the maximum value $\#_{\min}(c)$. In our PhD place example, TU Berlin, Oxford University, and MIT are all winners under Maximin voting. Clearly, winner determination using Maximin voting can be done in time polynomial in the input size. The *Maximin* concept originates from decision theory [Wal49,Sni08]. There are many other names for this voting protocol. For instance, Fishburn [Fis77] called it *Condorcet procedure* and Young [You77] used the name *Minimax function.*

We conclude this section with a remark on the relation between scoring protocols and Condorcet-related protocols. Condorcet [dC85] argued that there are elections whose Condorcet winner is not elected by any scoring protocol that awards more points to the first ranked candidate than to the second ranked one, and so forth; for example, this holds true for Borda voting [BF02]. The following example is due to Brams and Fishburn [BF02, Section 9.3] and shall illustrate this phenomenon. Suppose that there is an election on three candidates, a, b, and c, with seven votes cast as follows:

three votes $a \succ b \succ c$,
two votes $b \succ c \succ a$,
one vote $b \succ a \succ c$, and
one vote $c \succ a \succ b$.

The Condorcet winner of this election is a. She beats both b and c by 4-to-3. However, any scoring protocol assigning strictly more points to a candidate placed 2nd than to a candidate placed 3rd makes b win. Indeed, Moulin [Mou91] showed that no positional scoring protocol fulfills the Condorcet principle.

3.3 Further Voting Protocols

In this section, we introduce two more commonly used voting protocols which require several stages to aggregate votes. We also discuss one additional issue concerning the election of multiple winners.

Plurality with Runoff. This voting protocol consists of two rounds. In the first round, it orders the candidates according to the number of votes in which they rank first; all candidates but the first two in this new order are eliminated from the original votes. In case that two or more candidates are tied to pass the first round, Conitzer et al. [CRX09] argued that a candidate c is a winner if and only if there exists a way to break ties in all steps such that c wins. In this survey, we adopt a specific tie-breaking rule: Let C_1 be the set of candidates that have the highest number of first positions, and let C_2 be the set of candidates that have the second-highest number of first positions.

- If $|C_1| = 1$ and $|C_2| = 1$, or $|C_1| = 2$, then go to the second round.
- If $|C_1| = 1$ and $|C_2| \geq 2$, then the candidate $c \in C_2$ who has the highest number of second positions stays. If $|C_1| \geq 3$, then the two candidates among C_1 with the highest numbers of second positions pass the first round. For both cases ($|C_1| = 1 \wedge |C_2| \geq 2$ or $|C_1| \geq 3$): If there are more than two candidates to pass the first round, then for tie-breaking the number of third positions is used, and so on. If, however, after $m - 1$ steps, still more than two candidates are tied, then all these candidates pass the first round.

In the second round, Plurality voting is applied to the input votes restricted to the candidates that pass the first round to elect a winner.

The second round can be omitted if in the first round there is a candidate who ranks first in more than half of the votes.

In our PhD place example, TU Berlin safely passes to the second round. However, ETH Zurich and MIT each rank first in one vote, and second in no votes, so we have to consider the votes where they rank third. MIT ranks third in two votes but ETH Zurich in only one vote, so MIT can stay for the second round. After eliminating the other candidates, TU Berlin and MIT are tied 2-to-2 in the second round, so they both are co-winners.

Variations of Plurality With Runoff voting are widely used in the presidential elections of many countries (such as Austria, Brazil, and France). It is criticized

for its so-called *no-show paradox* [Mou91], which means that sometimes it may be advantageous not to submit your vote. Let us see an example to better understand this paradox. Suppose that there are 100 votes on the candidates, a, b, and c, with

30 votes $a \succ c \succ b$,
41 votes $b \succ a \succ c$, and
29 votes $c \succ b \succ a$.

The winner according to Plurality with Runoff is b. However, if two of the voters who favor a abstain, then in the first round a will be eliminated and c beats b by 59-to-41 in the second round. While this does not make candidate a win, these votes do prefer candidate c to candidate b.

Single Transferable Voting (STV). To select a single winner, STV deletes the candidates ranked first in the fewest votes. This procedure is repeated until a candidate ranks first in more than half of the *restricted votes*–the votes without deleted candidates. By deleting some candidates, some originally lower ranked candidates can be *transferred* to a higher position. STV can take up to $m - 1$ stages with m being the total number of candidates. This happens if in each stage no candidate ranks first in more than half of the restricted votes. Note that if there are only three candidates, then STV for the single winner case is equivalent to Plurality with Runoff voting, and, hence, suffers from the same "no-show paradox".

When using STV in our PhD place example, Oxford University and Tsinghua University will be first deleted from the votes: No vote ranks Oxford University or Tsinghua University as the first candidate. Then every candidate ranking behind Oxford University or Tsinghua University in the original votes will be transferred to a higher position:

Parameterized complexity:	B $\succ$ M $\succ$ Z
Salary:	Z $\succ$ M $\succ$ B
English usage:	M $\succ$ B $\succ$ Z
Cultural activities:	B $\succ$ Z $\succ$ M

In the next stage, we delete MIT and ETH Zurich from the remaining votes. Finally, the only candidate remaining, that is, TU Berlin, is the winner according to STV.

STV with some modifications is often used in political elections, for instance in Australia, Ireland, and New Zealand.

Obviously, the winner determination problem for Plurality with Runoff or STV can be solved in time polynomial in the input size.

Multi-Winner Protocols. Multi-winner elections come into play whenever one has to elect an assembly whose members need to be authorized to take decisions

on behalf of the society. Hence, for a multi-winner voting protocol, it is important to elect an assembly (winner set) that represents the society adequately. Although the protocols stated above can be easily modified to return a set of winners, for all of them except for STV this does arguably not lead to an appropriate choice of winners [BF02,LB11]. An alternative way is based on the concept of "misrepresentation". Basically, in this model, each vote can assign a misrepresentation value to every candidate. The set of winners is selected from the candidates such that the total misrepresentation is minimized.

Borda voting is a natural example for a misrepresentation function: Every vote assigns a misrepresentation value of zero to his favorite candidate, a value of one to his second choice, a value of two to the third choice, and so on.

One natural approach for selecting winners is to choose a set of, say, k winning candidates such that the sum of misrepresentation values is minimized (*minimum sum*); another way is to minimize the maximum misrepresentation (*minimax*) [BEH$^+$10,BF02]. In both cases, in the model suggested by Chamberlin and Courant [CC83] every candidate can represent an unlimited number of votes, that is, within a selected assembly a vote is always represented by an assembly member for whom its misrepresentation value is minimal. Since this may lead to the situation that different assembly members represent different numbers of votes, Chamberlin and Courant suggested to use weights as a way out. In contrast, the model suggested by Monroe [Mon95] requires that every assembly member represents about the same number of votes, that is, at most $\lceil n/k \rceil$ and at least $\lfloor n/k \rfloor$ for n votes and k winners.

Unfortunately, all four problem variants resulting from combining Chamberlin and Courant's as well as Monroe's approach with "minimax" or "minimum sum" optimization are already NP-hard for the basic Borda misrepresentation function [BSU11,LB11,PRZ08].

Parameterized complexity analysis with respect to the parameters "number of winners", "total misrepresentation value", "number of voters", and "number of candidates" has been started only recently [BSU11].

4 Kemeny Voting

In this section, we provide a case study on different parameterizations of the voting problem Kemeny Score (which was mentioned in Section 3.2). The optimization problem behind Kemeny Score can also be seen as a natural combinatorial median finding problem: Given a multiset of rankings, find a ranking that is "closest" to the given rankings. Here, the distance measure is the so-called Kendall-Tau distance. Let (C,V) be an election and let l be a ranking over C. Then, the *score* of l is defined as

$$\sum_{v \in V} \text{KT-dist}(v, l),$$

where KT-dist(v, l) denotes the *Kendall-Tau distance*. The Kendall-Tau distance between $v \in V$ and l is defined as

$$\text{KT-dist}(v, l) := \sum_{\{a,b\} \subseteq C} d_{v,l}(a, b),$$

where $d_{v,l}(a, b)$ is 0 when v and l rank a and b in the same relative order, and 1, otherwise. Formally, the corresponding decision problem is defined as follows:

Kemeny Score

Input: An election (C, V) and a non-negative integer k.

Question: Is there a ranking with score at most k?

A *Kemeny consensus* l^* is a ranking with minimum score. The *Kemeny score* of a given election is the score of l^*.

The Kemeny score of our PhD place example (see Section 3) is sixteen. For instance, the ranking B $\succ$ O $\succ$ M $\succ$ Z $\succ$ T and the ranking O $\succ$ M $\succ$ B $\succ$ T $\succ$ Z each forms a Kemeny consensus. There are altogether eighteen different Kemeny consensuses. The reason is that most candidate pairs are tied 2-to-2 and both relative orderings of these two candidates contribute the same to the score. Every Kemeny consensus for our PhD place example realizes the cheaper relative ordering for all four non-tied candidate pairs (see Table 2 or Figure 1 in Section 3.2).

For small examples like this, a Kemeny consensus is easy to find. However, in practice one often has to deal with larger and more complicated instances. Kemeny Score is NP-hard, but in some applications exact solutions are required. Here, parameterized algorithmics comes into play. In the remainder of this section, we overview recent research concerning the parameterized complexity of Kemeny Score (see also Table 3).

4.1 Input and Output Parameterizations

Three parameters directly appear in the problem definition of Kemeny Score. The parameters "number n of votes" and "number m of candidates" are given by the input. The parameter "Kemeny score k" is given by the solution of the problem.

Number n of Votes. Kemeny Score is NP-hard even for elections with only four votes [DKNS01a,DKNS01b]. This means that there is no hope for fixed-parameter tractability with respect to the parameter "number of votes". To the best of our knowledge, NP-hardness for Kemeny Score with a constant odd number of votes is still open. On the contrary, NP-hardness for Kemeny Score with an unbounded odd number of votes has been shown by Bartholdi et al. [BTT89b].

Table 3. Parameterized complexity of Kemeny Score and two of its generalizations. In case of fixed-parameter tractability results, we only state the exponential parts of the corresponding running times if provided in the corresponding papers. "NP-h" means NP-hard. Results marked by (♣) follow from [DKNS01a,DKNS01b], (♢) follow from [KS10], (♠) follow from [MRS09], and (♡) follow from [BGKN11]. The remaining results are provided in [BFG$^+$09]. Note that "?" means that the corresponding case remains open whereas "—" means that the corresponding parameter does not apply to the problem.

	Kemeny Score	with ties	incomplete votes
# votes n	NP-h for $n = 4$ (♣)	NP-h for $n = 4$ (♣)	NP-h for $n = 4$ (♣)
# candidates m	2^m	2^m	2^m
Kemeny score k	$2^{\mathcal{O}(\sqrt{k})}$ (♢)	1.76^k	$k! \cdot 4^k$
max. range r_m	32^{r_m}	$(3r_m + 1)! \cdot 2^{3r_m+1}$	—
avg. range r_a	NP-h for $r_a \geq 2$	NP-h for $r_a \geq 2$	—
max. KT-dist d_m	$2^{\mathcal{O}(\sqrt{d_m})}$ (♢)	$(6d_m + 2)! \cdot 2^{6d_m+2}$	NP-h for $d_m = 0$
avg. KT-dist d_a	$2^{\mathcal{O}(\sqrt{d_a})}$ (♢)	$2^{\mathcal{O}(d_a^2)}$ (♡)	NP-h for $d_a = 0$
$\overline{d} := k/n$	$2^{\mathcal{O}(\sqrt{\overline{d}})}$ (♢)	$2^{\mathcal{O}(\overline{d}^2)}$ (♡)	NP-h for $\overline{d} = 0$
above guarantee	FPT (♠)	?	?

Number m of Candidates. Kemeny Score becomes fixed-parameter tractable for the parameter "number of candidates". This is easy to see: Try all possible $m!$ rankings over C, compute the corresponding scores, and check whether the minimum score is at most k. Note that, given an election with n votes and m candidates, one can compute the score of any ranking in $\mathcal{O}(n \cdot m \log m)$ time [KT06].

By a dynamic programming approach, one can improve the exponential part of the running time from $m!$ to 2^m [BFG$^+$09,RS07]. The basic idea behind the dynamic programming is to compute a Kemeny consensus for the elections restricted to subsets of candidates: The dynamic programming table contains a Kemeny consensus for each subset of candidates. We compute the entries for all subsets of size s beginning with $s = 1$. Then, we increase s until we get the entire candidate set. The initialization of the table is easy, because elections with only one candidate induce exactly one ranking. The recurrence behind the dynamic programming is as follows. Consider the computation of an entry for a subset $C' \subseteq C$. For each $c \in C'$, compute the score of the ranking beginning with c and concatenated with the Kemeny consensus for $C' \setminus \{c\}$ obtained from the dynamic programming table. Now, the entry for C' is a ranking with minimum score.

Kemeny Score k. The Kemeny score measures the "distance of the solution from the input votes". The following two simple data reduction rules lead to a problem kernel with at most $2k$ votes and at most $2k$ candidates [BFG$^+$09]. Herein, we call a pair of candidates $\{a, b\}$ *conflict pair* if there is one vote with "$a \succ b$" and another vote with "$b \succ a$" in the election.

Rule 1 *Delete every candidate that is not involved in any conflict pair.*

Rule 2 *If there are more than k identical votes, then return "yes" if the score of one of them is at most k; otherwise, return "no".*

The problem kernel obtained through Rules 1 and 2 already shows fixed-parameter tractability of Kemeny Score with respect to the parameter k. This can be improved by considering the conflict pairs. In this way, one obtains bounded search-tree algorithms which are much faster than an $\mathcal{O}^*((2k)!)$-time[7] brute-force strategy or an $\mathcal{O}^*(2^{2k})$-time dynamic programming algorithm operating on the problem kernel. First, observe that the number of conflict pairs is at most k for every yes-instance [BFG$^+$09]. A search-tree which decides for each conflict pair which of both orderings appears in a Kemeny consensus has size $\mathcal{O}(2^k)$. Considering "conflict triples" one obtains an improved algorithm with running time $\mathcal{O}(1.53^k + m^2 \cdot n)$ [BFG$^+$09]. Further refined search-tree algorithms lead to search-tree sizes of $\mathcal{O}^*(1.403^k)$ [Sim09]. Besides search tree algorithms, further approaches were considered in the literature to solve Kemeny Score—yielding sub-exponential time fixed-parameter algorithms with respect to the parameter k [ALS09,FFL$^+$10,KS10].

4.2 Structural Parameterizations

Depending on the voting protocols used, voting problems provide a large amount of interesting structural parameters. For Kemeny Score, we discuss the parameters "maximum range r_m of candidate positions", "average range r_a of candidate positions", "maximum KT-distance d_m between the input votes", and "average KT-distance d_a between the input votes". All four parameters are illustrated with the help of our PhD place example (see Table 1) in Figure 2. This section will be concluded by a brief discussion of an "above average parameterization" for Kemeny Score.

The parameters "maximum range r_m of candidate positions" and "average range r_a of candidate positions" both use a common concept called the *range* of a candidate. The *range* of a candidate c is defined as one plus the difference between her best and worst position.

Maximum Range r_m of Candidate Positions. The maximum range of candidate positions is the range of the candidate who has the maximum range. It seems plausible that instances with a bounded range of candidate positions are easier to solve. Indeed, using dynamic programming, one can solve Kemeny Score in $\mathcal{O}(32^{r_m} \cdot (r_m^2 \cdot m + r_m \cdot m^2))$ time [BFG$^+$09].

[7] The notation $\mathcal{O}^*(.)$ is similar to $\mathcal{O}(.)$, but only states the superpolynomial part of the running time.

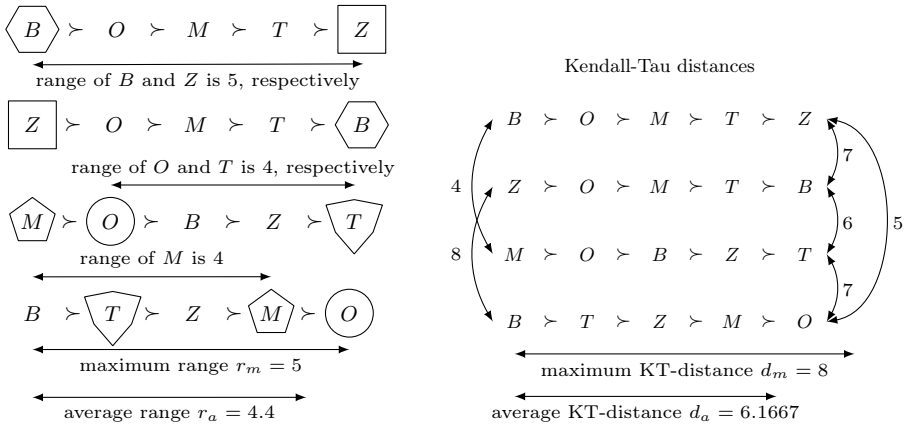

Fig. 2. Illustration of structural parameters for Kemeny Score. On the left we have our four votes from the PhD place example where the range of each candidate, that is, the difference between the worst and the best position is highlighted. The first vote is also one possible Kemeny consensus. On the right, we depict the KT-distances between every pair of input votes (written as labels on the arcs).

Average Range r_a of Candidate Positions. Analogously to the maximum range, the average range r_a of candidate positions is the average range of all candidates. A small maximum range indicates instances which are easy to solve, while instances with a small average range of candidate positions remain hard. Even for instances with $r_a = 2$ Kemeny Score remains NP-hard [BFG$^+$09]: Given a Kemeny Score instance (C, V, k), one can construct an equivalent instance with average range 2 by adding $|C|^2$ many new candidates and putting them at the end of every vote (for each vote in the same order). Each new candidate has a range of one and hence the average range is at most

$$\frac{|C| \cdot |C| + |C|^2}{|C|^2 + |C|} \leq 2.$$

Based on the Kendall-Tau distance, we discuss three further parameterizations.

Average KT-Distance d_a. The average KT-distance is formally defined as

$$d_a := \sum_{v,w \in V} \frac{\text{KT-dist}(v, w)}{n(n-1)}.$$

It measures "the average amount of variety in the votes". In the first fixed-parameter algorithm with respect to parameter d_a [BFG$^+$09], the authors basically observed that in every Kemeny consensus each candidate may only occur in a fixed range of positions whose size is bounded by d_a. Based on this observation, there is a dynamic programming algorithm that solves Kemeny Score

in $\mathcal{O}(16^{d_a} \cdot (d_a^2 \cdot m + d_a \cdot m \log m \cdot n) + n^2 \cdot m \log m)$ time. This was improved by Simjour [Sim09] who developed a search tree algorithm with running time $\mathcal{O}^*(5.833^{d_a})$. Furthermore, Karpinski and Schudy [KS10] developed a subexponential fixed-parameter algorithm with running time $2^{\mathcal{O}(\sqrt{d_a})} + n^{\mathcal{O}(1)}$.

Besides fixed-parameter algorithms with respect to the parameter average KT-distance, data reduction rules were developed whose performance guarantee depends on the average KT-distance. Although no problem kernel in the classical sense is known, the currently best upper bound on the number m of candidates is linear in the average KT-distance [BBN10]. This is achieved by applying polynomial-time data reduction. Note that the non-existing bound on the number n of votes does not harm too much, since Kemeny Score is fixed-parameter tractable with respect to m. More precisely, it was shown that exhaustive application of the following simple rule already yields a "partial problem kernel" [BBN10,BGKN11].

Rule 3 *If there is a candidate c such that there is no other candidate c' with $1/4 \cdot |V| \leq \#(c, c') \leq 3/4 \cdot |V|$, then remove c (and adjust the allowed score accordingly[8]).*

Exhaustive application of Rule 3 yields an equivalent instance of Kemeny Score with at most $16/3 \cdot d_a$ candidates [BBN10].

Maximum KT-Distance d_m. Clearly, fixed-parameter tractability for d_a also implies fixed-parameter tractability for the parameter "maximum KT-distance d_m between two input votes". However its potentially larger values (compared to average KT-distance) allow for improvements in the algorithm. With slight modifications in the search tree algorithm for Kemeny Score parameterized by d_a, one can solve Kemeny Score in $\mathcal{O}^*(4.829^{d_m})$ time [Sim09]. Note that the subexponential fixed-parameter algorithm due to Karpinski and Schudy [KS10] for d_a also works for d_m.

Parameterizations Above Average $k_{\min}$. Mahajan and Raman [MR99] introduced "parameterization above guaranteed values" as a general form of parameterization. For Kemeny Score, a guaranteed value is a lower bound on the Kemeny score k, for instance

$$k_{\min} := \sum_{\{a,b\} \subseteq C} \min\{\#(a, b), \#(b, a)\}.$$

This is an obvious lower bound for k, because it is simply the sum of the minimum contributions for each candidate pair. A natural question is to parameterize above this guaranteed lower bound, that is, by the parameter "$k - k_{\min}$". Fixed-parameter tractability with respect to $(k - k_{\min})$ for Kemeny Score is implied by a parameter-preserving reduction from Kemeny Score to a weighted variant of Directed Feedback Vertex Set [MRS09].

[8] For each candidate c' with $\#(c', c) > 3/4 \cdot |V|$, decrease the score by $\#(c, c')$; otherwise, decrease the score by $\#(c', c)$.

4.3 Ties and Incomplete Votes

In this section, we briefly discuss results obtained for two generalizations of Kemeny Score. In the first generalization, we modify our election model such that candidates may also be ranked equally, that is, we allow for ties. The second generalization is to allow for incomplete votes, that is, considering partial orders instead of linear orders (see Section 2 for a formal definition of incomplete votes). In contrast to the parameterization by "number of candidates", which can also be used for both generalizations more or less without any modification (compare with Section 4.1), for most other parameterizations the situation changes when we consider the more general models.

Kemeny Score with Ties. In the Kemeny Score generalization Kemeny Score with Ties [Ail10,HSV05] one additionally allows that two candidates in a vote are ranked equally. Now, the term $d_{v,w}(a,b)$ expressing the contribution of the candidate pair $\{a,b\}$ to the KT-distance between two votes v and w is defined as

$$d_{v,w}(a,b) = \begin{cases} 2 & \text{if } (a \succ b \text{ in } v \text{ and } b \succ a \text{ in } w) \text{ or } (b \succ a \text{ in } v \text{ and } a \succ b \text{ in } w), \\ 0 & \text{if } a \text{ and } b \text{ are ordered in the same way in } v \text{ and } w, \text{ and} \\ 1 & \text{otherwise.} \end{cases}$$

There are slightly different models for the consensus of an election with ties in the literature: Hemaspaandra et al. [HSV05] allowed that the consensus can also have ties, while Ailon [Ail10] defined the consensus as permutation of candidates (without ties).

Betzler et al. [BFG+09] analyzed the parameterized complexity of Kemeny Score with Ties for the setting of Hemaspaandra et al. [HSV05]. With similar approaches as described in Section 4.1, one obtains a search tree of size $\mathcal{O}(1.76^k)$ as well as a polynomial-size problem kernel with respect to the parameter "Kemeny score k" [BFG+09].

Concerning structural parameters such as maximum range r_m or average range r_a, one has to be careful when ties are allowed. Betzler et al. [BFG+09] used an intuitive concept where, similarly to the classical Kemeny Score, the range is defined as the difference between the best and the worst position. However, to make these positions in a vote with ties uniquely determined, the best position of a candidate is defined as the minimum number of candidates that are better than her and her worst position is defined as the maximum number of candidates that are better or equally ranked.

It is not obvious how to transfer the results for structural parameterizations with classical Kemeny Score to Kemeny Score with Ties. However, fixed-parameter tractability with respect to the parameter maximum range r_m of candidate positions can be obtained by an approach similar to the dynamic programming algorithm for classical Kemeny Score with respect to r_m [BFG+09]. Furthermore, when extending the problem by additionally assigning weights to candidates, the dynamic programming approach also covers the parameterization with maximum KT-distance d_m. The maximum range of candidate positions

is bounded by $2 \cdot d_m$ for instances with candidate weights [BFG+09]. Finally, a partial kernelization with respect to the parameter average KT-distance can be transferred to Kemeny Score with Ties [BGKN11].

Kemeny Score with Incomplete Votes. In the Kemeny Score generalization Kemeny Score with Incomplete Votes [DKNS01a], the given votes are not required to be permutations of the entire candidate set, but of candidate subsets.[9] In contrast to the votes, the Kemeny consensus is a permutation of all candidates. As a consequence, the term $d_{v,w}(a,b)$ expressing the contribution of the candidate pair $\{a,b\}$ to the KT-distance between two votes v and w is adjusted to

$$d_{v,w}(a,b) := \begin{cases} 0 & \text{if } \{a,b\} \not\subseteq C_v \text{ or } \{a,b\} \not\subseteq C_w \text{ or } v \text{ and } w \text{ agree on } a \text{ and } b, \\ 1 & \text{otherwise,} \end{cases}$$

where C_v contains the candidates occurring in vote v.

Since one can have non-trivial instances without "conflict pairs", the branching approach for classical Kemeny Score does not apply to the parameterization with the Kemeny score k when we allow for incomplete votes. However, by a parameterized reduction to Weighted Feedback Arc Set, one obtains fixed-parameter tractability [BFG+09].

As to structural parameterizations, defining the range of candidate positions does not make sense. Furthermore, Kemeny Score with Incomplete Votes remains NP-hard even if the maximum KT-distance d_m between two input votes is zero, that is, there is no hope for fixed-parameter tractability with respect to the parameters average KT-distance d_a and maximum KT-distance d_m [BFG+09].

5 Types of Voting Problems

In this section, we review a number of voting problems and account for their computational complexity, both standard and parameterized. We start with the most immediate question in Section 5.1: Can a candidate win an election under a given voting protocol? In later sections we deal with more subtle voting problems, often rendering the considered problems already hard for scoring protocols. In Section 5.2, we consider possible winner determination in case of incomplete votes (partial orders instead of linear orders), then move on to the related problem of manipulating elections (Section 5.3), and eventually study questions of bribery, control, and optimal lobbying in Sections 5.4, 5.5, and 5.6.

5.1 Winner Determination

The most basic computational task in voting is the determination of a winner using a given voting protocol $\mathcal{E}$. Alternatively, we can ask whether a given candidate is a winner of an election under voting protocol $\mathcal{E}$:

[9] This only yields a subset of all possible partial orders.

Table 4. Parameterized complexity results for computationally hard winner determination problems. Considered parameters are the number m of candidates, the number n of votes, and the number k of modifications. For Dodgson Score, k denotes the number of swaps; for Young Score, k denotes the number of remaining votes, while for Dual Young Score k denotes the number of deleted votes. The fixed-parameter tractability results for parameter m follow from integer linear programming formulations [BTT89b,You77] and Lenstra's result on integer linear programming with a fixed number of variables [Len83]. Results marked by (△) follow from [BTT89b,You77], by (♣) from [FJL$^+$10], by (♡) from [BGN10], and by (♠) from [RSV03].

Parameter	Dodgson Score	Dual Young Score	Young Score
m	FPT (△)	FPT (△)	
n	W[1]-hard (♣)	FPT ($\mathcal{O}^*(2^n)$) (△)	
k	FPT ($\mathcal{O}^*(2^k)$) (♡)	W[2]-complete (♡)	W[2]-complete (♡, ♠)

$\mathcal{E}$ Winner Determination

Input: An election (C, V) and a distinguished candidate $p \in C$.

Question: Does p win the election under voting protocol $\mathcal{E}$?

A voting protocol having nice properties but for which one cannot compute a winner in reasonable time is not useful in practice. While for some voting protocols such as positional scoring protocols, the computation of a winner can be easily achieved in polynomial time, for other voting protocols the computation of a winner is NP-hard. Famous voting protocols with NP-hard winner determination are listed in the survey by Chevaleyre et al. [CELM07]. This includes the voting protocols proposed by Banks, Dodgson, Kemeny, Slater, and Young.[10]

In Table 4, we list parameterized complexity results for Dodgson Score, Dual Young Score, and Young Score with respect to several parameterizations.

5.2 Possible & Necessary Winner

Possible Winner. In standard voting scenarios, one typically assumes that voters provide their preferences as linear orders. To determine a winner, the given linear orders are aggregated according to a voting protocol. However, in many realistic settings, the voters may provide partial orders only [KL05]. This directly leads to the Possible Winner problem which, given a set of incomplete votes, asks whether a specific candidate can still become a winner if one extends the votes to

[10] Banks [Ban85] and Slater [Sla61] are two comparison-based voting protocols. They both work on the majority graph of a given election (see Definition 5) and are closely related to graph problems restricted to tournaments [CH00,Woe03]. A tournament is a directed graph with exactly one arc between any two vertices. See Woeginger [Woe03] and Hudry [Hud04] for the computation of Banks winners, and Charon and Hudry [CH00] and Conitzer [Con06] for the computation of Slater winners.

linear orders (see Section 2 for detailed information on partial orders and their extensions).

Let us go back to the student from Section 3 who decides to do his PhD research at the TU Berlin. At the enrollment, it happens that there is a My Favorite Professor evaluation among four candidate professors: Prof. Bosch (B), Prof. Geiger (G), Prof. Hertz (H), and Prof. Zuse (Z). [11] Until now, only sixteen students have participated in the evaluation. Five of them like Prof. Bosch as much as Prof. Geiger (in the subsequent example expressed by {B, G}), followed by Prof. Hertz and then by Prof. Zuse. Another five students favor Prof. Hertz over Prof. Zuse, followed by Prof. Bosch, while ranking Prof. Geiger as the least-liked candidate. The remaining six students prefer Prof. Geiger and Prof. Zuse to Prof. Hertz. Their least favorite professor is Prof. Bosch. The current state of the evaluation is as follows:

Five students with $\{\text{B, G}\} \succ \text{H} \succ \text{Z}$,
five students with $\text{H} \succ \text{Z} \succ \text{B} \succ \text{G}$, and
six students with $\{\text{G, Z}\} \succ \text{H} \succ \text{B}$.

Obviously, the preferences of the students are not all linear orders. Hence, instead of determining the best ranked professor, we are interested in determining the *possible winners* of the election. For instance, to ask whether Prof. Hertz is a possible winner under Borda voting in the above evaluation is to determine whether there are extensions of students' preferences such that Prof. Hertz becomes a winner. In our example, such extensions exist: If the first five students submit the linear oder $\text{B} \succ \text{G} \succ \text{H} \succ \text{Z}$, three of the last six students submit $\text{G} \succ \text{Z} \succ \text{H} \succ \text{B}$ and the other three students submit $\text{Z} \succ \text{G} \succ \text{H} \succ \text{B}$, then Prof. Hertz (26 points) becomes a winner under Borda voting. However, to determine whether a distinguished candidate is a possible winner in an election using Borda voting is NP-complete [XC11] (also see Table 5).

Formally, Possible Winner for a given voting protocol $\mathcal{E}$ is defined as follows:

$\mathcal{E}$ Possible Winner

Input: An election (C, V) with the multiset $V = \{v_1, \ldots, v_n\}$ of incomplete votes represented as partial orders on C, and a distinguished candidate $p \in C$.

Question: Is there a multiset $V' = \{v'_1, \ldots, v'_n\}$ of votes over C, such that each vote v'_i extends v_i and p wins the election (C, V') under voting protocol $\mathcal{E}$?

The motivation behind Possible Winner is that it might be impossible for the voters to provide a complete ranking because, for instance, the set of candidates is too large. Another reason can be that not all voters might have given their rankings yet during the aggregation process, or new candidates

[11] Historical note: Only Hans Geiger and Gustav Hertz were professors at TU Berlin whereas Carl Bosch and Konrad Zuse were students at TU Berlin.

Table 5. Summary of (parameterized) complexity results for the NP-complete [XC11] Possible Winner using several common voting protocols. Parameters considered are "the number m of candidates", "the number n of votes", "the total number s of undetermined candidate pairs", and "the maximal number u of undetermined candidate pairs in a vote". Note that the fixed-parameter tractability results for parameter s hold for all voting protocols whose Winner Determination problem can be solved in time polynomial in the input size. Fixed-parameter tractability results for parameter m are again due to Lenstra's result on integer linear programming with a fixed number of variables [Len83]. Results marked with (♣) come from [XC11], those with (♡) come from [BHN09]. Note that "?" means that the corresponding case remains open and para-NP-c means that the problem remains NP-complete even for constant parameter values.

Parameter	Borda	k-Approval	Copeland$^\alpha$
m	FPT	FPT	FPT
n (♡)	para-NP-c	para-NP-c	?
s (♡)	$\mathcal{O}^*(1.82^s)$	$\mathcal{O}^*(2^s)$	$\mathcal{O}^*(2^s)$
u (♣)	para-NP-c	para-NP-c	para-NP-c

might be introduced after some voters already have given their rankings (see also [CLM+11]). Moreover, one often has to deal with incomplete votes due to two or more candidates not being comparable, because of lack of information or other reasons. Hence, the study of incomplete voting profiles is natural and essential.

Again, we survey standard and parameterized computational complexity results for Possible Winner under various voting protocols. Notably, although Winner Determination problems are straightforward for (most) scoring protocols, Possible Winner is already computationally hard for simple scoring protocols such as k-Approval. Table 5 lists the results together with references to the literature.

Due to the way how k-Approval assigns points to candidates, two further structural parameters immediately pop up in the study of k-Approval Possible Winner: The "number k of approvals in each vote" and the "number $k' = m - k$ of disapprovals in each vote" with m being the total number of candidates. However, k-Approval Possible Winner is already NP-complete for any constant number $k \geq 2$ [XC11]. This motivates a multivariate complexity analysis [Fel09,Nie10] with respect to the combined parameter number n of votes and number k (k') of candidates to whom a voter gives one (zero) point. Parameterized complexity results for k-Approval Possible Winner are summarized in Table 6.

There are many interesting open questions concerning $\mathcal{E}$ Possible Winner. In the following we just mention a few:

– Until now, existing studies [BD10,BHN09,Wal07,XC11] on Possible Winner consider only scoring protocols as well as some comparison-based proto-

Table 6. Parameterized complexity results of k-Approval Possible Winner, where t denotes the number of incomplete votes in an election, k denotes the number of ones assigned in the k-Approval voting, while k' denotes the number of zeros assigned in the k-Approval voting. Results marked with (♣) come from [XC11], while (♡) marks results from [Bet10b].

Parameter	Results	Remarks
k (♣)	NP-complete	For any fixed $k \geq 2$
(t, k') (♡)	FPT	$\mathcal{O}^*(\min\{2^{t^2k'},\ 2^{tk'} \cdot (tk')^{k'}\})$
(t, k) (♡)	FPT	Super-exponential kernel

cols that are computationally efficient (polynomial time solvable) for Winner Determination, since if the $\mathcal{E}$ Winner Determination is computationally hard, then $\mathcal{E}$ Possible Winner is also computationally hard. It would be interesting to see whether the fixed-parameter tractability results for Kemeny Score, Dodgson Score, Young Score, or Dual Young Score still hold for Possible Winner where incomplete votes are given.

– As we have seen in Table 5, the parameter "total number s of undetermined candidate pairs" leads to fixed-parameter tractability; however, s may be very large for some scenarios. On the contrary, Possible Winner is already NP-complete for Borda, k-Approval, and Copeland$^\alpha$ voting even if the maximal number u of undetermined candidate pairs in a vote is a constant [XC11]. This motivates further parameterizations concerning incomplete votes of an election. For example, it would be interesting to know whether Possible Winner is fixed-parameter tractable with respect to the parameter "average/maximum number of undetermined candidate pairs in which a candidate is involved".

Necessary Winner. Finally, we mention in passing that, in addition to the Possible Winner problem, there is also the Necessary Winner problem, which asks whether a given distinguished candidate is a winner in *all* extensions of the given votes. As a rule of thumb, it appears that the Necessary Winner problem is computationally easier than the Possible Winner problem. For example, Necessary Winner can be solved in polynomial time for scoring protocols as well as some other protocols such as Plurality with Runoff, Maximin voting, and Bucklin[12], while Possible Winner is NP-complete for these voting protocols [XC11]. We refer to the literature [KL05,PRVW11,XC11] for more details.

[12] Bucklin voting is a hybrid voting protocol. In a nutshell, it combines k-Approval with Majority voting. Majority voting is similar to Plurality with the additional constraint that the candidate who has a score of more than half of the number of votes wins. See Xia and Conitzer [XC11] for a definition and more in-depth explanation.

5.3 Manipulation

Manipulation is a voting scenario where a *coalition of voters* casts their votes in an insincere way such that they end up better off than voting honestly. We illustrate such a situation with the help of the My Favorite Professor example. Suppose that the election has

five students with $B \succ G \succ H \succ Z$,
five students with $H \succ Z \succ B \succ G$,
three students with $G \succ Z \succ H \succ B$, and
three students with $Z \succ G \succ H \succ B$.

Under Borda voting, Prof. Hertz (26 points) wins the election. Suppose that the last three students in the above election know the votes of all other thirteen students.[13] They want to make their favorite candidate, Prof. Zuse, win the election. Hence they form a *coalition* and try to *manipulate* the election result by casting their own votes contrary to their actual preferences. Although they all prefer Prof. Geiger and Prof. Hertz to Prof. Bosch, by submitting

two votes $Z \succ G \succ B \succ H$ and
one vote $Z \succ B \succ G \succ H$,

together with the other thirteen votes, Prof. Zuse will indeed become the Borda winner with 25 points instead of Prof. Hertz with 23 points.

For manipulation, we assume that the voters of the coalition know about all the votes of the sincere voters. The coalition uses *strategic voting* to achieve their goal of letting their favorite candidate win. Formally, the decision problem Manipulation for any voting protocol $\mathcal{E}$ is defined as follows:

$\mathcal{E}$ Manipulation

Input: An election (C, V), a coalition size $k \in \mathbb{N}$ encoded in unary alphabet, and a distinguished candidate $p \in C$.

Question: Is there a multiset V' of at most k votes on C such that p is the winner according to $\mathcal{E}$ in $(C, V \cup V')$?

The $\mathcal{E}$ Manipulation problem can be considered as a special case of the $\mathcal{E}$ Possible Winner problem: The non-manipulative votes are linear orders and the manipulative votes are totally empty. Hence, any hardness result on $\mathcal{E}$ Manipulation is also valid for $\mathcal{E}$ Possible Winner.

A voting protocol is *strategy-proof* if manipulation is never beneficial for any voter or coalition of voters. A famous result of Gibbard and Satterthwaite [Gib73,Sat75] states that a *resolute*[14], *surjective*[15], and strategy-proof

[13] This is rarely the case in practice. However, it allows for a worst-case analysis.
[14] A voting protocol is *resolute* if there is always exactly one winner for an election.
[15] A voting protocol is *surjective* if every candidate has a chance of winning.

voting protocol is dictatorial[16]. Bartholdi et al. [BTT89a] suggested using computational hardness to "resist" manipulations in an election: The idea is that if a voting protocol can be manipulated in principle, but it is computationally intractable to decide whether it is possible to cast the votes to achieve a desired result, then this voting protocol is unlikely to be manipulated in practice. In particular, Bartholdi et al. [BO91,BTT89a] focused on the special case of having a coalition of size one: After obtaining polynomial-time solvability results for manipulation under a set of common voting protocols including Plurality, Borda, Maximin and Copeland [BTT89a], Bartholdi and Orlin [BO91] showed that STV Manipulation is NP-hard even for a single manipulator. However, Conitzer et al. [CSL07] showed fixed-parameter tractability with respect to "the number m of candidates" for STV Manipulation with a coalition of size one. The corresponding algorithm runs in $\mathcal{O}^*(1.62^m)$ time. Recent studies [BNW11,DKNW11] show that Borda Manipulation is already NP-complete for a coalition of size two. When parameterized by "the number of candidates", Borda Manipulation is fixed-parameter tractable [BHN09]. A further parameter is derived from so-called "instance tightness", again yielding fixed-parameter tractability [BNW11].

Since $\mathcal{E}$ Manipulation is a special case of $\mathcal{E}$ Possible Winner, some open computational hardness questions stated in Section 5.2 can also be transformed to the context of $\mathcal{E}$ Manipulation.

For STV Manipulation with one manipulator, there is a fixed-parameter algorithm with respect to "the number of candidates" [CSL07]. Naturally, it would be interesting to know whether this also holds for two or more manipulators.

5.4 Bribery

As the name suggests, *bribery* is another attack on elections, where the briber "pays" some voters to have them change their votes in order to reach a desired outcome [FHH09]. Typically, the briber has a budget. The basic question with respect to bribery is whether the briber can achieve his goal without exceeding his budget.

There are different settings of bribery: Besides varying prices for different voters, one relaxes the notion of votes by allowing arbitrary relations instead of linear orders [Fal08,FHHR09b]. In addition, there are more fine-grained models such as paying for specific operations. For instance, in Swap Bribery [EFS09], one is only allowed to perform *swaps* of two neighboring candidates in a vote. Formally, a swap in some vote $v \in V$ is a triple (v, c_1, c_2) where $\{c_1, c_2\} \subseteq C, c_1 \neq c_2$. Applying a swap (v, c_1, c_2), that is, exchanging the positions of c_1 and c_2 in the vote v, is *admissible* when c_1 and c_2 are neighbors in v. A sequence of swaps is called admissible when the application of the swaps in the given ordering is admissible in each case. The decision problem is defined as follows:

[16] *Dictatorial* means that there exists a voter who always decides what the outcome of an election shall be.

$\mathcal{E}$ Swap Bribery

Input: An election (C, V), a distinguished candidate $p \in C$, a budget $\beta \in \mathbb{N}$, and a cost function $c : V \times C \times C \to \mathbb{N}$.

Question: Is there an admissible sequence Γ of swaps with $\sum_{s \in \Gamma} c(s) \leq \beta$ such that p wins the election under voting protocol $\mathcal{E}$ after having applied the swaps as given by Γ?

For our My Favorite Professor example (see Section 5.3 for the complete list of student votes), we already know that Prof. Hertz wins under Borda voting. Suppose that a fan of Prof. Bosch knows all the votes of the students. His goal is to make Prof. Bosch win the election via swap bribery. A single swap costs one Euro. However, he is only willing to pay at most four Euros. Now the question is whether, without exceeding his budget, the fan of Prof. Bosch can bribe some students and let them *swap* neighboring candidates in their votes such that Prof. Bosch wins the election. If he bribes the three students with identical original vote G $\succ$ Z $\succ$ H $\succ$ B and one student whose original vote is Z $\succ$ G $\succ$ H $\succ$ B, and lets them each swap the two neighboring candidates H and B in their votes, then he can make B (Prof. Bosch with 26 points) win the election. Each of the four swaps has a cost of one Euro, so the budget (four Euros) is not exceeded.

Table 7 shows some computational complexity results for k-Approval Swap Bribery. Classical complexity results are given by Elkind et al. [EFS09], while the parameterized results are provided by Dorn and Schlotter [DS12]. It should be mentioned that $\mathcal{E}$ Possible Winner can be seen as a special case of $\mathcal{E}$ Swap Bribery, where the price of any determined candidate pair is one, swapping two *undetermined* neighboring candidates[17] has cost zero, and the budget is zero. So hardness results on $\mathcal{E}$ Possible Winner are also valid for $\mathcal{E}$ Swap Bribery for some restricted scenarios.

Restricting the allowed operations such that each swap must involve the distinguished candidate leads to Shift Bribery [EFS09]. As for the parameterized complexity analysis of this scenario, we refer to a recent study by Schlotter et al. [SEF11].

In *microbribery* [FHHR09b], a briber can invert the relative order of any two candidates in a vote for a given price. Typically, this leads to votes which are no longer linear orders. For example, inverting the relative order of a and c in a vote $a \succ b \succ c$ results in three pairwise comparisons: $a \succ b$, $b \succ c$, and $c \succ a$.

Elkind and Faliszewski [EF10a] initiated research on another aspect of bribery which concerns *campaign management*. There, bribing voters means, for instance, investing in advertisement for a specific candidate. They argued that such kind of campaign can strongly influence the outcome of an election. This has applications in political elections or product marketing. Schlotter et al. [SEF11] studied both classical and parameterized complexity regarding two specific cases of campaign management, *shift bribery* and *support bribery*, for several voting protocols.

[17] Recall that two neighboring candidates are called *undetermined* if they are not comparable in the incomplete vote.

Table 7. Parameterized complexity results of k-Approval Swap Bribery [DS12]. The parameters are "the budget β", "the number n of votes", "the number m of candidates", and "the number k of approved candidates in a vote". Note that k-Approval Swap Bribery is already NP-complete for $k = 2$ due to its close relationship to Possible Winner [BD10,DS12,EFS09]. With respect to the parameter m, the fixed-parameter tractability result holds not only for k-Approval but indeed for a wide range of voting protocols including Copeland$^\alpha$ and Maximin [DS12].

Parameter	Results	Remarks
β	W[1]-hard for $n = 1$	Reduction from Multi-Colored Clique
k	W[1]-hard	Reduction from Clique
m	FPT for constant k	Integer linear programming
n	FPT for constant k	Color-coding
(β, n)	FPT	Kernel with $n^2\beta^2$ candidates and $n^2\beta$ votes
(β, n, k)	FPT	Kernel with $(n + k)\beta$ candidates and $n^2\beta$ votes

Destructive bribery, that is, using bribery to prevent one candidate from winning, is NP-hard for Copeland$^\alpha$ and Maximin voting [FHH09,FHH11,FHHR09b]. Until now, parameterized complexity aspects in this context seem to be unexplored, presenting good opportunities for new research.

5.5 Control

To *control* an election, an external agent, somewhat misleadingly called the *chair* in the literature, can change the *election structure* to reach certain goals. For example, a typical question is whether the chair can make his favorite candidate a winner by deleting some candidates. Going back to our My Favorite Professor example, using Copeland$^{0.5}$ also results in selecting Prof. Hertz as the most favorite professor (2 points). If a fan of Prof. Bosch who wants to influence the election is in the election committee and somehow manages to disqualify Prof. Zuse from the election, then all three remaining candidates become (co)-winners (1 point) according to Copeland$^{0.5}$.

Actually, there are many different types of control including *adding* or *deleting* candidates or votes [BTT92]. Furthermore, one distinguishes between *constructive control* (CC), where the chair aims at making a distinguished candidate a winner, and *destructive control* (DC), where the chair wants to prevent a distinguished candidate from winning [HHR07]. In the following, we define the $\mathcal{E}$ Constructive Control Via Adding Candidates ($\mathcal{E}$ CC-AC).

$\mathcal{E}$ CC-AC

Input: Two disjoint sets C, D of candidates, a multiset V of votes over $C \cup D$, a distinguished candidate $p \in C$, and a non-negative integer k.

Question: Is there a subset $D' \subseteq D$ of candidates with $|D'| \leq k$ such that p is the winner in the election $(C \cup D', V)$ according to voting protocol $\mathcal{E}$?

Three more types of constructive control problems, that is, via deleting candidates, via adding votes, and via deleting votes, can be defined analogously: For the case of adding votes, we are given a multiset of votes from which we can select additional votes in order to change the outcome of an election. The decision problems of destructive control via adding or deleting candidates or votes can be defined accordingly: Instead of making the distinguished candidate a winner, destructive control aims at precluding the distinguished candidate from winning.

The investigation of the computational complexity of control problems goes back to Bartholdi et al. [BTT92]. Since then, there has been a series of publications [BTT92,FHHR09b,HHR07] which provides a complete picture of the classical computational complexity for 22 basic types of control. These papers cover standard voting systems such as Plurality, Condorcet, and Copeland$^\alpha$ for all rational values of α in the range of $[0, 1]$. For example, one of the voting protocols that can be used to determine the winner of an election in time polynomial in the input size and is NP-hard for all standard types of constructive control is Copeland$^{0.5}$ [FHHR09b].

Hemaspaandra et al. [HHR09] showed that so-called *hybrid* elections can lead to stronger resistance results for electoral control. Further work looks into control for two specific hybrid systems combining Approval voting and systems based on linear preferences [EF10b,ENR09,EPR10,ER10].

A closely related problem introduced by Elkind et al. [EFS10a] is cloning, where one only allows for adding candidates that are "similar"[18] to one of the existing candidates. Moreover, Chevaleyre et al. [CLM$^+$11] investigated the question whether a candidate can become a winner by adding "arbitrary" candidates.

Recently, Faliszewski et al. [FHH11] introduced the extended scenario of "multi-mode control attacks", that is, the chair is allowed to use various kinds of attacks like deleting candidates and adding votes simultaneously.

Table 8 lists some parameterized complexity results for eight different kinds of control: Constructive Control via Adding Candidates (CC-AC), Constructive Control via Deleting Candidates (CC-DC), Constructive Control via Adding Votes (CC-AV), Constructive Control via Deleting Votes (CC-DV), and their destructive control (DC) versions.

Table 9 shows some results on control of elections employing Copeland$^\alpha$. Considered parameters are the number m of candidates and the number n of votes. Furthermore, there are also results concerning parameterized complexity for Copeland$^\alpha$, $\alpha = 1$, with respect to non-standard parameters like "feedback arc set size of the majority graph" [BBNU11].

We conclude this section with a few interesting research directions. The parameterized complexity of many scoring protocols, such as Borda, seems to be unexplored. Multi-mode control as proposed by Faliszewski et al. [FHH11] seems a natural candidate for a multivariate complexity study. For instance, the problem whether a distinguished candidate can win by deleting k candidates and adding k' votes under Copeland^1 is NP-hard [FHH11]; it is an open question

[18] Here, a candidate c_1 is called similar to another candidate c_2 if for each vote, candidates c_1 and c_2 have the same relative position to any other candidate.

Table 8. Control-related (parameterized) complexity results. All W-hardness results are with respect to the output parameter. For example, Plurality CC-AC is W[2]-hard with respect to the number of added candidates. Results marked with (♠) come from [BTT92], those with (♢) come from [HHR07], those with (♣) come from [LFZL09], those with (□) come from [LZ10], results marked with (♡) come from [BU09], those marked with (△) come from [FHHR09b], those marked with (◯) come from [EF10b,EFPR11,ER10], and those marked with (▽) come from [BGN10]. The W[2]-completeness result of Dual Young Score holds for Condorcet CC-DV because they are equivalent. Any entry labeled "P" means polynomial-time solvability. "/" means either that we are not aware of any meaningful parameterized complexity results or that it is irrelevant. For example, in Condorcet CC-AC, the chair can never make a non-winning candidate win the election by adding some additional candidates [BTT92]. W[t]-h stands for W[t]-hard with $t = 1$ or $t = 2$; W[2]-c stands for W[2]-complete. Recall that Bucklin voting is a hybrid voting protocol which combines k-Approval with Majority voting [EF10b,XC11]. Fallback [BS09] voting combines Bucklin with Approval voting. Here, Approval voting, slightly different from k-Approval, allows each voter to approve of an arbitrary number of candidates.

	Plurality	Condorcet	Maximin	Copeland$^\alpha$	Bucklin/Fallback
CC-AC	W[2]-h (♠)	/	W[2]-h (□)	W[2]-c (♡)	W[2]-h (◯)
CC-DC	W[2]-h (♡)	P (♠)	/	W[2]-c (♡)	W[2]-h (◯)
CC-AV	P (♠)	W[1]-h (♣)	W[1]-h (□)	/	W[2]-h (◯)
CC-DV	P (♠)	W[2]-c (▽)	W[1]-h (□)	/	W[2]-h (◯)
DC-AC	W[2]-h (♢)	P (♢)	/	P (△)	W[2]-h (◯)
DC-DC	W[1]-h (♡)	/	/	P (△)	W[2]-h (◯)
DC-AV	P (♢)	P (♢)	W[1]-h (□)	/	P (◯)
DC-DV	P (♢)	P (♢)	W[1]-h (□)	/	P (◯)

Table 9. Parameterized complexity results on control of elections using Copeland$^\alpha$. We use (♣) to denote the results from [FHHR09b] and (♡) to denote the results from [BU09]. Results on control by adding (deleting) candidates with a bounded number of added (deleted) candidates as well as on control by adding (deleting) votes with a bounded number of added (deleted) votes follow from brute-force enumeration of all possible subsets of candidates of votes. For the case of candidate control with a bounded number of votes, the fixed-parameter algorithms are based on Lenstra's integer linear programming result [Len83].

	AC	DC	AV	DV
# candidates m	FPT (♣)	FPT (♣)	FPT (♣)	FPT (♣)
# votes n	NP-c (♡)	NP-c (♡)	FPT (♣)	FPT (♣)

whether this NP-hard problem is fixed-parameter tractable with respect to the combined parameter (k, k').

5.6 Lobbying

Sometimes we do not only vote on one but on multiple issues at the same time. A corresponding voting procedure can be very simple: Approve or disapprove of each issue. Formally, a *multi-issue election* for m issues and n voters is an $n \times m$ binary matrix

$$W = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,m} \end{pmatrix} \in \{0,1\}^{n \times m}.$$

An entry $w_{i,j}$ of W represents voter i's opinion on issue j: 0 stands for disapproval; 1 stands for approval.

Given a multi-issue election and desired outcomes for each issue, Christian et al. [CFRS07] studied how hard it is to "lobby" some voters optimally, that is, to persuade the minimum number of voters to change their votes such that each issue has a majority of voters with values equal to the desired outcome. The formal definition of Optimal Lobbying is as follows:

Optimal Lobbying

Input: A multi-issue election $W \in \{0,1\}^{n \times m}$, a non-negative integer k, and a size m target vector $x \in \{0,1\}^m$.

Question: Can W be transformed into a new matrix $W' \in \{0,1\}^{n \times m}$ by editing entries in at most k different rows such that for each column j there is a strict majority of rows with value x_j?

If Optimal Lobbying can be shown to be computationally intractable, then potential attackers may not succeed in influencing the outcome of a multi-issue election via lobbying in reasonable time. This is what Bartholdi et al. [BTT89b] and Faliszewski et al. [FHH10] meant by using complexity to protect elections. But what about more restricted scenarios, that is, what about parameterized complexity analysis? To the best of our knowledge, Christian et al. [CFRS07] started the parameterized complexity analysis of voting problems concerning lobbying in multi-issue elections. They showed that Optimal Lobbying is W[2]-complete with respect to the number of votes to be changed. The idea of the proof of this result is shown in Section 6.6. Erdély et al. [EFG+09] further extended Optimal Lobbying to a probabilistic setting and, in particular, provided several results on fixed-parameter tractability and W[2]-completeness. Finally, we remark that Optimal Lobbying is fixed-parameter tractable with respect to the number of issues, since it can be easily transformed into an integer linear program with a fixed number of variables. See also Section 6.4 for more details.

6 Parameterized Techniques

In this section, we overview different techniques for investigating fixed-parameter tractability which already have been successfully applied in the area of voting. We

start with some techniques for designing fixed-parameter algorithms and close with a general technique to obtain intractability results. Each technique will be accompanied by an example. Although these standard techniques "cover" most results so far, there are further approaches to obtain fixed-parameter tractability results in the context of voting [ALS09,FFL$^+$10,KS10].

6.1 Search Trees

A search tree algorithm identifies a "small subset" of the input instance such that at least one part of the subset is part of a solution. Then, it branches over all possible parts of this small subset to fix it as part of the solution. This procedure is repeated in a recursive manner until the whole solution has been found. In the context of fixed-parameter algorithms, the identification of the subset is done in polynomial time and the search tree size is bounded by some function only depending on the parameter.

For instance, in the case of Kemeny Score (see Section 4.1) a simple search tree algorithm identifies as small subset the two possible orderings a candidate pair can have in the solution. Since one can show that for at most k many candidate pairs, where k denotes the Kemeny score of the election, the ordering is not yet clear, the search tree size is bounded by $\mathcal{O}(2^k)$.

Next, we briefly discuss a search tree approach that applies to Possible Winner for arbitrary voting protocols [BHN09]. For every undetermined candidate pair, say $\{a, b\}$ from incomplete vote v, branch into the following two possible cases: Either add $a \succ b$ to v or add $b \succ a$ to v. If an option violates the transitivity of v, then discard the corresponding branch in the search tree. The search tree size is at most $\mathcal{O}(2^s)$, where s is the total number of undetermined candidate pairs, implying that for every voting protocol with polynomial-time winner determination, Possible Winner is fixed-parameter tractable with respect to the parameter s.

Similarly to the improved search tree for Kemeny Score (see Section 4.1), one gains a refined fixed-parameter algorithm for Possible Winner through considering undetermined triples instead of pairs combined with a network flow construction [BHN09].[19] As a consequence, for a specific class of scoring protocols, including k-Approval and Borda, Possible Winner can be decided in $O(1.82^s \cdot (nm^2 + s^2))$ time, where s is the total number of undetermined pairs.

6.2 Kernelization

Problem Kernels. Recall from Section 2 that a problem kernel can be seen as an equivalent instance whose size is bounded by a function in the parameter and which can be computed by polynomial-time preprocessing (so-called data reduction rules) [Bod09,GN07].

[19] Indeed, techniques based on network flows are used in several other voting contexts to derive polynomial-time solvability for special cases or as part of a fixed-parameter algorithm (see, for example, [BD10,BHN09,DS12,FHHR09b]).

For instance, by exhaustive application of Rules 1 and 2 from Section 4.1 one gets a problem kernel with at most $2k$ votes and at most $2k$ candidates for Kemeny Score.

Dorn and Schlotter [DS12] developed a kernelization algorithm that constructs a problem kernel with $\mathcal{O}(n^2 \cdot \beta)$ votes and $\mathcal{O}(n^2 \cdot \beta^2)$ candidates for k-Approval Swap Bribery, where n denotes the number of votes and β denotes the budget. Kernelization algorithms have also been developed for k-Approval Possible Winner: Problem kernels for Possible Winner with respect to the combined parameters (t, k) as well as (t, k') have been obtained by data reduction rules [Bet10a,Bet10b], where t denotes the number of incomplete votes, k denotes the number of one-point positions, and k' denotes the number of zero-point positions. For (t, k') there is a kernel with $\mathcal{O}(t \cdot {k'}^2)$ candidates and $\mathcal{O}(t^2 \cdot {k'}^2)$ votes, while for (t, k) there is a superexponential-size kernel that shows fixed-parameter tractability.

No Polynomial Kernel. A natural question for a fixed-parameter tractable problem is: How small can a corresponding problem kernel be? In particular, can we expect to derive a polynomial-size kernel for every fixed-parameter tractable problem? To answer this, Bodlaender et al. [BDFH09] and Fortnow and Santhanam [FS11] introduced a general framework; also see the surveys by Bodlaender [Bod09] and Misra et al. [MRS11]. The basic idea is that if a parameterized version of an NP-complete problem has a so-called "composition algorithm", then it does not admit a polynomial-size problem kernel, under some widely believed complexity assumptions. Furthermore, such lower-bound results can be transferred to other problems by so-called "polynomial parameter transformations" [BTY11].

For instance, Fellows et al. [FJL$^+$10] showed that Dodgson Score with respect to the parameter number of swaps (see also Section 3.2 for the corresponding definition) is unlikely to admit a polynomial-size kernel. This result is obtained by a polynomial parameter transformation from Small Universe Hitting Set which is known to be unlikely to have a polynomial-size problem kernel [DLS09].

Partial and Turing Kernels. There are cases where it seems hard to bound the whole size of the instance by a polynomial function in the parameter, but it is possible to bound only one dimension[20] of the input by such function. Here, the concept of *partial kernelization* [BGKN11] comes into play. For instance, for Kemeny Score one has a partial kernel with respect to the average KT-distance d_a, that is, one can construct equivalent instances with at most $16/3 \cdot d_a$ candidates, but the number of votes is unbounded [BBN10].

Furthermore, it could also be possible that one cannot find a problem kernel, but one is able to compute polynomially many instances whose sizes are bounded by some function in the parameter and the original instance is a yes-instance if and only if one of the new instances is a yes-instance. This leads to the

[20] In case of voting, for example the number of candidates or the number of votes are two natural dimensions.

concept of *Turing kernelization*, or to be more specific, disjunction truth-table kernelization [FFL$^+$09,Lok09]. We are not aware of Turing kernelization results in voting.

Both, partial kernels and Turing kernels provide (similarly to the classical kernel concept) the possibility of obtaining fixed-parameter algorithms.

6.3 Dynamic Programming

The key idea of dynamic programming is to solve a problem by solving subproblems and to combine overlapping solutions to find an overall solution. Dynamic programming tries to avoid multiple computation of the same subsolution by storing it in a so-called *dynamic programming table*. It is a standard technique in mathematics and computer science and in the design of fixed-parameter algorithms [Nie06]. Often leading to very efficient algorithms, a typical bottleneck of dynamic programming is its memory consumption which may also be exponential in the parameter.

For instance, with dynamic programming one can solve Kemeny Score in $O^*(2^m)$ time (see Section 4.1). However, also the space requirement is $O^*(2^m)$.

A further example is Dodgson Score. Using dynamic programming it can be solved in $O(2^k \cdot nk \cdot nm)$ time [BGN10], where k denotes the number of swaps.

6.4 Integer Linear Programming

As one of the most popular techniques for problem solving, (integer) linear programming[21] is also useful for classification and algorithm design in the context of parameterized algorithmics [Nie06]. A famous result of Lenstra [Len83] implies that a problem is fixed-parameter tractable when it can be solved by an integer linear program where the number of variables is upper-bounded by a function solely depending on the parameter.

Bartholdi et al. [BTT89b] developed an integer linear program to solve Dodgson Score and gave a running time bound based on Lenstra's result. They did not explicitly state this, but this shows fixed-parameter tractability for Dodgson Score with respect to the parameter number m of candidates. The corresponding integer linear program is shown in Figure 3. Note that it computes the Dodgson score of a specific candidate.

Although solvability by an integer linear program with a bounded number of variables implies fixed-parameter tractability, there is by far no guarantee for practically efficient algorithms. Indeed, due to a huge exponential function in the number of variables being part of the running time bound, Lenstra's [Len83] result is basically for classification only.

There are several similar fixed-parameter tractability results with respect to the parameter number of candidates for control problems [FHHR09b], Possible Winner [BHN09], and Swap Bribery [DS12] for various voting protocols.

[21] See, for example, Matoušek and Gärtner [MG06] for a general introduction to linear programming.

$$\min \sum_{i,j} j \cdot x_{i,j} \text{ subject to}$$
$$\forall i \in \tilde{V} : \sum_{j} x_{i,j} = N_i$$
$$\forall y \in C : \sum_{i,j} e_{i,j,y} \cdot x_{i,j} \geq d_y$$
$$x_{i,j} \geq 0$$

Fig. 3. Integer linear program determining the Dodgson score of candidate c. Here, C denotes the set of candidates, $\tilde{V}$ denotes the set of ranking types (that is, the set of votes where identical votes appear only once), N_i denotes the number of votes of type i, $x_{i,j}$ denotes the number of votes with rankings of type i for which candidate c will be moved upwards by j positions, $e_{i,j,y}$ is 1 if the result of moving candidate c by j positions upward in a ranking of type i is that c gains an additional vote against candidate y, and 0 otherwise. Furthermore, d_y is the deficit of c with respect to candidate y, that is, the minimum number of votes that c must gain against y to defeat her in a pairwise comparison. If c already defeats y, then $d_y = 0$. For more details see Bartholdi et al. [BTT89b]. Altogether, the integer linear program contains at most $m \cdot m!$ variables $x_{i,j}$ and at most $m! + m$ non-trivial constraints, where m denotes the number of candidates.

Furthermore, Dorn and Schlotter [DS12] convey without details that their result concerning Swap Bribery can be transferred to problems like Optimal Lobbying and Manipulation under some specific voting protocols as well.

6.5 Color-Coding

Alon et al. [AYZ95] introduced *color-coding* as a randomized algorithm for solving some types of graph problems. Recently, Dorn and Schlotter [DS12] used it to show the fixed-parameter tractability of k-Approval Swap Bribery with respect to the number of votes for constant k (see Section 5.4). Here we sketch the idea behind their randomized fixed-parameter algorithm and how it employs color-coding.

Let I be an instance of k-Approval Swap Bribery consisting of an election (C, V) with $|C| = m$ and $|V| = n$, a distinguished candidate $d \in C$, a budget $\beta \in \mathbb{N}$, and a cost function $c : V \times C \times C \to \mathbb{N}$. Instance I is a yes-instance if and only if there is a sequence Γ of swaps with $\sum_{s \in \Gamma} c(s) \leq \beta$, and d wins after the swaps in Γ have been performed. We denote the votes after having performed Γ as $V^\Gamma = \bigcup_{v \in V} v^\Gamma$.

A candidate is *relevant* with respect to Γ if it receives at least one point in V^Γ. Let $C^{\text{rel}}(\Gamma)$ be the set of candidates relevant with respect to Γ. Since each of the first k candidates of a vote receives one point according to k-Approval, and since there are at most nk relevant candidates, we can identify each vote through a *vote pattern* which is a size-k subset of $\{1, \ldots, nk\}$. It should represent

the set of the first k candidates. An *election pattern* $P = (p_1, \ldots, p_n)$ is an n-tuple of vote patterns. There are $\binom{nk}{k}^n < (nk)^{nk}$ such election patterns. If the distinguished candidate d wins the bribed election, then we can assume that $d \in C^{\mathrm{rel}}(\Gamma)$. We also require that d represents the number 1. Thus, an election pattern P is called *successful* if 1 appears at least as frequently as any other number between 2 and nk in P.

The basic idea of the algorithm is as follows: For each successful election pattern $P = (p_1, \ldots, p_n)$, we color each candidate (except for candidate d) randomly with one of the colors of $\bigcup_{p \in P} p \setminus \{1\}$; d has color 1. If I is a yes-instance, then with probability of at least $(nk-1)^{1-nk}$ we can find in $(nk)^{nk} \cdot \mathcal{O}(m^{k+1})$ randomized time a sequence Γ of swaps[22] with the following properties: each relevant candidate in C^{rel} has a different color (while p has color 1), the colors of relevant candidates of vote ${v_i}^{\Gamma} \in V^{\Gamma}$ form the vote pattern $p_i \in P$, and the budget is not exceeded. Trying all possible successful election patterns, the algorithm takes a total of $(nk)^{2nk}\mathcal{O}(m^{k+1})$ randomized time. Note that using nk-perfect hash functions [AYZ95], one gains a deterministic fixed-parameter algorithm with respect to the parameter n for constant k.

6.6 Parameterized Intractability

There are several voting problems where computational intractability can be desirable for a protocol. Intractability in terms of parameterized complexity means W[t]-hardness for some integer $t \geq 1$ (see Section 2). Without going into the details of the theory, the main message is that W[t]-hard problems are not fixed-parameter tractable under several widely believed complexity assumptions (including the Exponential Time Hypothesis [IPZ01]). We present a simple parameterized reduction showing W[2]-hardness in what follows; refer to the textbooks [DF99,FG06,Nie06] for general accounts.

To the best of our knowledge, the first W-hardness result result for a computational social choice problem is due to Christian et al. [CFRS07]. They considered the problem Optimal Lobbying as defined in Section 5.6.

Parameterized Reduction for Optimal Lobbying. The idea behind the proof of parameterized intractability for Optimal Lobbying is to describe a parameterized reduction (see Section 2) from the W[2]-complete problem Dominating Set to Optimal Lobbying [CFRS07] (see Section 5.6). Dominating Set asks, given an undirected graph $G = (V, E)$, whether there exists a size-k subset of vertices $V' \subseteq V$ such that every vertex is either from V' or has a neighbor in V'. Such a vertex subset is called *dominating set*.

The construction of the Optimal Lobbying instance works as follows.[23] The matrix W is an extension of the adjacency matrix of G. First take the

[22] See Dorn and Schlotter [DS12] for the detailed algorithm to find the swap sequence Γ when a successful election pattern and a feasible coloring are given.

[23] Note that the original construction works with interchanged roles for 1 and 0. This slight modification allows for a compact way of only presenting the idea.

adjacency matrix of G and add one additional *selection column* filled with 1s. Then, add $|V| - 2k + 1$ *dummy rows* filled with 0s. We call the original $|V|$ rows *vertex rows* and the original $|V|$ columns *vertex columns*. Finally, for each vertex column i, flip $|V| - k - N[i] + 1$ entries in the dummy rows from 0 to 1, where $N[i]$ denotes the number of neighbors of the vertex corresponding to column i. The target vector x (see Section 5.6) has a 0 in each of the $|V| + 1$ positions.

Now, we have the following situation. First, consider the selection column. The number of 1s exceeds the number of 0s by $2k - 1$, that is, k of the graph rows must be chosen in the solution. Now, consider the vertex columns. For each column the number of 1s exceeds the number of 0s only by 1. Hence, there is a majority for 0s if and only if the chosen vertex rows correspond to vertices forming a dominating set. In analogy, every dominating set implies such a solution.

Roughly speaking, this means that Optimal Lobbying remains intractable even if the number of voters to influence is small.

7 Discussion and Future Challenges

So far, the consideration of problems from algorithmic graph theory prevails in parameterized complexity studies. The impact of parameterized complexity analysis, however, strongly hinges on its high potential to explain, to predict, and to engineer computational complexity. The "computational complexity landscape" of problems arising in real-world applications needs a more fine-grained consideration than classical (one-dimensional) complexity analysis delivers. Thus, in 2008, two issues of *The Computer Journal* (Volume 51, Numbers 1 and 3 edited by Rod G. Downey, Michael R. Fellows, and Michael A. Langston) cover applications of parameterized complexity analysis in bioinformatics, computational geometry, artificial intelligence, constraint satisfaction, data bases, and cognitive modelling. Clearly, this list is far from being complete and deserves further additions. With this survey, we try to overview and promote the research on parameterized (and multivariate) complexity of voting problems, a subfield of the strongly growing area of computational social choice. Indeed, voting problems seem to be a particularly fruitful ground of (future) parameterized complexity analysis for at least three reasons:

- many NP-hard voting problems have simple and clear combinatorial definitions;
- many voting problems carry very natural structural parameters such as the number of candidates or the number of votes, with application scenarios where these parameter values are anticipated to be small;
- it is very natural and sometimes forcing to search for *exact* solutions.

The parameterized complexity analysis of voting problems leaves numerous challenges for future research. Some of these have been indicated in the preceding sections. Moreover, there are many NP-hard voting problems that have not yet been studied from a parameterized complexity perspective.

We conclude with a few more specific research questions and directions concerning the parameterized computational complexity of NP-hard voting problems (refer to a recent PhD thesis [Bet10a] for additional material).

- A central parameter in voting problems is the number of candidates (equivalently, alternatives). There is a number of fixed-parameter tractability results for this parameter [Bet10a,BHN09,DS12,EFS10b,FHHR09b] relying on integer linear programming and exploiting Lenstra's result [Len83] for a fixed number of variables. It would be highly desirable to replace these results by direct combinatorial algorithms with more efficient running times.
- There are numerous results in the theory of voting [ASS02,ASS10] providing structural properties of specific voting systems. These might be exploited for spotting interesting parameters in voting problems. For instance, Elkind et al. [EFS10b] explored and exploited "distance rationalizability" to show fixed-parameter tractability results. Pini et al. [PRVW11] used "independence of irrelevant alternatives" to even gain polynomial-time solvability for a restricted Possible Winner voting problem.
- From an algorithmic point of view, the established parameterized technique iterative compression [RSV04,GMN09] seems widely unexplored. Moreover, there are only few kernelization results in voting (see Section 6.2). Notably, it seems difficult to come up with kernelizations for the parameter "number of votes". Hence, this calls for combined parameters in the spirit of multivariate algorithmics [Fel09,Nie10] or the development of partial kernelizations [BBN10,BGKN11] where only one input dimension is reduced.
- Many fixed-parameter tractability results in voting (as in algorithmic graph theory) are of theoretical nature only. It remains a general task to improve the efficiency of these results and to finally arrive at implementations and experiments in the spirit of algorithm engineering. For instance, algorithm engineering for computing Kemeny scores revealed that data reduction (based on partial kernelization results) combined with (integer) linear program solvers leads to practically relevant results [BBN10].
- Voting is an ideal playground for multivariate algorithmics [Fel09,Nie10]. In particular, for identifying (structural) parameters to exploit, it seems worthwhile to explore many of the NP-hardness proofs for voting problems. For instance, a proof showing NP-hardness for Kemeny Score with only four votes [DKNS01a,DKNS01b] reveals that in order to work, it requires a high average KT-distance between the votes, making this a plausible parameter. Hence, "deconstructing intractability" [KNU11,Nie10] appears particularly beneficial in case of voting problems.
- Voting provides numerous challenging combinatorial problems which are not about graphs.[24] However, directed graph problems pop up in many voting problems. For instance, there are close connections (also employed

[24] In August 2011, Michael R. Fellows and Frances A. Rosamond organized at Charles Darwin University, Australia, the *Workshop–Parameterized Complexity: Not About Graphs (NAG)* in order to stimulate more parameterized complexity research beyond graph problems.

for proving NP-hardness results) between voting and problems on tournaments [BBS11,KS10,Woe03] or control in voting and vertex deletion problems on directed graphs [BBNU12,BU09,FHHR09b].

Voting problems are highly attractive from a parameterized complexity analysis perspective; this survey hopefully helps to attract more parameterized research in this fruitful and important area. Be invited!

Acknowledgements

We are grateful to Britta Dorn, Piotr Faliszewski, Jiong Guo, Matthias Mnich, Jörg Rothe, Ildikó Schlotter, and an anonymous referee for their numerous insightful remarks and their constructive advice.

References

AB09. Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009. 2

Ail10. Nir Ailon. Aggregation of Partial Rankings, p-Ratings, and Top-m Lists. *Algorithmica*, 57(2):284–300, 2010. 19

ALS09. Noga Alon, Daniel Lokshtanov, and Saket Saurabh. Fast FAST. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming*, volume 5555 of *LNCS*, pages 49–58. Springer, 2009. 16, 32

ASS02. Kenneth Joseph Arrow, Amartya K. Sen, and Kotaro Suzumura, editors. *Handbook of Social Choice and Welfare, Volume 1*. North-Holland, 2002. 3, 6, 38

ASS10. Kenneth Joseph Arrow, Amartya K. Sen, and Kotaro Suzumura, editors. *Handbook of Social Choice and Welfare, Volume 2*. North-Holland, 2010. 3, 6, 38

AYZ95. Noga Alon, Raphael Yuster, and Uri Zwick. Color-Coding. *Journal of the ACM*, 42(4):844–856, 1995. 35, 36

Ban85. J. S. Banks. Sophisticated Voting Outcomes and Agenda Control. *Social Choice and Welfare*, 1(4):295–306, 1985. 21

BBD09. Therese C. Biedl, Franz-Josef Brandenburg, and Xiaotie Deng. On the Complexity of Crossings in Permutations. *Discrete Mathematics*, 309(7):1813–1823, 2009. 1

BBN10. Nadja Betzler, Robert Bredereck, and Rolf Niedermeier. Partial Kernelization for Rank Aggregation: Theory and Experiments. In *Proceedings of 5th International Symposium on Parameterized and Exact Computation*, volume 6478 of *LNCS*, pages 26–37. Springer, 2010. 18, 33, 38

BBNU11. Nadja Betzler, Robert Bredereck, Rolf Niedermeier, and Johannes Uhlmann. On Making a Distinguished Vertex Minimum Degree by Vertex Deletion. In *Proceedings of the 37th International Conference on Current Trends in Theory and Practice of Computer Science*, volume 6543 of *LNCS*, pages 123–134. Springer, 2011. 29

BBNU12. Nadja Betzler, Robert Bredereck, Rolf Niedermeier, and Johannes Uhlmann. On Bounded-Degree Vertex Deletion Parameterized by Treewidth. *Discrete Applied Mathematics*, 160(1–2):53–60, 2012. 39

BBS11. Felix Brandt, Markus Brill, and Hans Georg Seedig. On the Fixed-Parameter Tractability of Composition-Consistent Tournament Solutions. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 85–90. AAAI Press, 2011. 39

BCE12. Felix Brandt, Vincent Conitzer, and Ulle Endriss. Computational Social Choice. In Gerhard Weiss, editor, *Multiagent Systems*. MIT Press, 2012. 1

BD10. Nadja Betzler and Britta Dorn. Towards a Dichotomy of Finding Possible Winners in Elections Based on Scoring Rules. *Journal of Computer and System Sciences*, 76(8):812–836, 2010. 23, 28, 32

BDFH09. Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On Problems Without Polynomial Kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009. 33

BEH+10. Dorothea Baumeister, Gábor Erdélyi, Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Computational Aspects of Approval Voting. In Jean-François Laslier and M. Remzi Sanver, editors, *Handbook on Approval Voting*, chapter 10, pages 199–251. Springer, 2010. 1, 13

Bet10a. Nadja Betzler. *A Multivariate Complexity Analysis of Voting Problems*. PhD thesis, Friedrich-Schiller-Universität Jena, 2010. 33, 38

Bet10b. Nadja Betzler. On Problem Kernels for Possible Winner Determination Under the k-Approval Protocol. In *Proceedings of the 35th International Conference on Mathematical Foundations of Computer Science*, volume 6281 of *LNCS*, pages 114–125. Springer, 2010. 24, 33

BF02. Steven Brams and Peter C. Fishburn. Voting Procedures. In Kenneth Joseph Arrow, Amartya K. Sen, and Kotaro Suzumura, editors, *Handbook of Social Choice and Welfare*, volume 1, pages 173–236. Elsevier, 2002. 10, 13

BFG+09. Nadja Betzler, Michael R. Fellows, Jiong Guo, Rolf Niedermeier, and Frances A. Rosamond. Fixed-Parameter Algorithms for Kemeny Rankings. *Theoretical Computer Science*, 410:4554–4570, 2009. 15, 16, 17, 19, 20

BGKN11. Nadja Betzler, Jiong Guo, Christian Komusiewicz, and Rolf Niedermeier. Average Parameterization and Partial Kernelization for Computing Medians. *Journal of Computer and System Sciences*, 77:774–789, 2011. 15, 18, 20, 33, 38

BGN10. Nadja Betzler, Jiong Guo, and Rolf Niedermeier. Parameterized Computational Complexity of Dodgson and Young Elections. *Information and Computation*, 208(2):165–177, 2010. 21, 30, 34

BHN09. Nadja Betzler, Susanne Hemmann, and Rolf 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, 2009. 23, 26, 32, 34, 38

BNW11. Nadja Betzler, Rolf Niedermeier, and Gerhard J. Woeginger. Unweighted Coalitional Manipulation Under the Borda Rule is NP-hard. In *Proceedings of 22nd International Joint Conference of Artificial Intelligence*, pages 55–60, 2011. 26

BO91. John J. Bartholdi III and James B. Orlin. Single Transferable Vote Resists Strategic Voting. *Social Choice and Welfare*, 8:341–354, 1991. 2, 26

Bod09. Hans L. Bodlaender. Kernelization: New Upper and Lower Bound Techniques. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation*, volume 5917 of *LNCS*, pages 17–37. Springer, 2009. 4, 32, 33

BS09. Steven Brams and M. Remzi Sanver. Voting Systems that Combine Approval and Preference. In Steven Brams, William V. Gehrlein, and Fred S. Roberts, editors, *The Mathematics of Preference, Choice, and Order: Essays in Honor of Peter C. Fishburn*, pages 215–237. Springer, 2009. 30

BSU11. Nadja Betzler, Arkadii Slinko, and Johannes Uhlmann. On the Computation of Fully Proportional Representation, 2011. Available at Social Science Research Network. 13

BTT89a. John J. Bartholdi III, Craig A. Tovey, and Michael A. Trick. The Computational Difficulty of Manipulating an Election. *Social Choice and Welfare*, 6(3):227–241, 1989. 2, 26

BTT89b. John J. Bartholdi III, Craig A. Tovey, and Michael A. Trick. Voting Schemes for Which It Can Be Difficult to Tell Who Won the Election. *Social Choice and Welfare*, 6(2):157–165, 1989. 2, 9, 14, 21, 31, 34, 35

BTT92. John J. Bartholdi, III, Craig A. Tovey, and Michael A. Trick. How Hard Is It to Control an Election? *Mathematical and Computer Modeling*, 16(8-9):27–40, 1992. 28, 29, 30

BTY11. Hans L. Bodlaender, Stéphan Thomassé, and Anders Yeo. Kernel Bounds for Disjoint Cycles and Disjoint Paths. *Theoretical Computer Science*, 412(35):4570–4578, 2011. 33

BU09. Nadja Betzler and Johannes Uhlmann. Parameterized Complexity of Candidate Control in Elections and Related Digraph Problems. *Theoretical Computer Science*, 410(52):5425–5442, 2009. 30, 39

CC83. John R. Chamberlin and Paul N. Courant. Representative Deliberations and Representative Decisions: Proportional Representation and the Borda Rule. *American Political Science Review*, 77(3):718–733, 1983. 13

CCDF97. Liming Cai, Jianer Chen, Rodney G. Downey, and Michael R. Fellows. Advice Classes of Parameterized Tractability. *Annals of Pure and Applied Logic*, 84:119–138, 1997. 4

CELM07. Yann Chevaleyre, Ulle Endriss, Jérôme Lang, and Nicolas Maudet. A Short Introduction to Computational Social Choice. In *Proceedings of the 33rd Conference on Current Trends in Theory and Practice of Computer Science*, volume 4362 of *LNCS*. Springer, 2007. 1, 21

CFRS07. Robin Christian, Mike Fellows, Frances Rosamond, and Arkadii Slinko. On Complexity of Lobbying in Multiple Referenda. *Review of Economic Design*, 11(3):217–224, 2007. 2, 31, 36

CH00. Irène Charon and Olivier Hudry. Slater Orders and Hamiltonian Paths of Tournaments. *Electronic Notes in Discrete Mathematics*, 5:60–63, 2000. 21

CLM+11. Yann Chevaleyre, Jérôme Lang, Nicolas Maudet, Jérôme Monnot, and Lirong Xia. New Candidates Welcome! Possible Winners with Respect to the Addition of New Candidates. *CoRR*, abs/1111.3690, 2011. 23, 29

Con06. Vincent Conitzer. Computing Slater Rankings Using Similarities among Candidates. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*, pages 613–619. AAAI Press, 2006. 21

Con10. Vincent Conitzer. Making Decisions Based on the Preferences of Multiple Agents. *Communications of the ACM*, 53:84–94, 2010. 1

Cop51. A. H. Copeland. A 'Resonable' Social Welfare Function, 1951. Mimeographed (University of Michigan Seminar on Application of Mathematics in Social Science). 8, 10

CRX09. Vincent Conitzer, Matthew Rognlie, and Lirong Xia. Preference Functions That Score Rankings and Maximum Likelihood Estimation. In *Proceedings*

of the 20th International Joint Conference on Artificial Intelligence, pages 109–115, 2009. 11

CSL07. Vincent Conitzer, Tuomas Sandholm, and Jérôme Lang. When Are Elections with Few Candidates Hard to Manipulate? *Journal of the ACM*, 54:1–33, 2007. 2, 26

dB81. Jean-Charles de Borda. Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences*, 1781. 7

dC85. Marie Jean Antoine Nicolas Caritat de Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Paris: L'Imprimerie Royale, 1785. 7, 9, 10

DF99. Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer Verlag, 1999. 2, 3, 4, 36

DKNS01a. Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank Aggregation Methods for the Web. In *Proceedings of the 10th International Conference on World Wide Web*, pages 613–622. ACM, 2001. 1, 9, 14, 15, 20, 38

DKNS01b. Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank Aggregation Revisited, 2001. Manuscript. 14, 15, 38

DKNW11. Jessica Davies, George Katsirelos, Nina Narodytska, and Toby Walsh. Complexity of and Algorithms for Borda Manipulation. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pages 657–662. AAAI Press, 2011. 26

DLS09. Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Incompressibility through Colors and IDs. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP'10)*, volume 5555 of *LNCS*, pages 378–389. Springer, 2009. 33

Dod76. Charles Dodgson. A Method of Taking Votes on More Than Two Issues. Pamphlet printed by the Clarendon Press, Oxford, and headed "not yet published", 1876. 8

DS12. Britta Dorn and Ildikó Schlotter. Multivariate Complexity Analysis of Swap Bribery. *Algorithmica*, 2012. Available electronically. 27, 28, 32, 33, 34, 35, 36, 38

DT11. Rodney G. Downey and Dimitrios M. Thilikos. Confronting Intractability via Parameters. *Computer Science Review*, 5(4):279–317, 2011. 4

EF10a. Edith Elkind and Piotr Faliszewski. Approximation Algorithms for Campaign Management. In *Proceedings of the 6th Workshop on Internet and Network Economics*, volume 6484 of *LNCS*, pages 473–482, 2010. 27

EF10b. Gábor Erdélyi and Michael R. Fellows. Parameterized Control Complexity in Bucklin Voting and in Fallback Voting. In *Proceedings of the 3rd International Workshop on Computational Social Choice*, pages 163–174, 2010. 29, 30

EFG$^+$09. Gábor Erdélyi, Henning Fernau, Judy Goldsmith, Nicholas Mattei, Daniel Raible, and Jörg Rothe. The Complexity of Probabilistic Lobbying. In *Proceedings of the 1st International Conference on Algorithmic Decision Theory*, volume 5783 of *LNCS*, pages 86–97. Springer, 2009. 31

EFPR11. Gábor Erdélyi, Michael R. Fellows, Lena Piras, and Jörg Rothe. Control Complexity in Bucklin and Fallback Voting. Technical report, arXiv:1103.2230, 2011. 30

EFS09. Edith Elkind, Piotr Faliszewski, and Arkadii Slinko. Swap Bribery. In *Proceedings of the 2nd International Symposium on Algorithmic Game Theory*, volume 5814 of *LNCS*, pages 299–310. Springer, 2009. 26, 27, 28

EFS10a. Edith Elkind, Piotr Faliszewski, and Arkadii Slinko. Cloning in Elections. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 768–773. AAAI Press, 2010. 29

EFS10b. Edith Elkind, Piotr Faliszewski, and Arkadii Slinko. On the Role of Distances in Defining Voting Rules. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 375–382, 2010. 38

ENR09. Gábor Erdélyi, Markus Nowak, and Jörg Rothe. Sincere-Strategy Preference-Based Approval Voting Fully Resists Constructive Control and Broadly Resists Destructive Control. *Mathematical Logic Quaterly*, 55:425–443, 2009. 29

EPR10. Gábor Erdélyi, Lena Piras, and Jörg Rothe. Control Complexity in Fallback Voting. Technical report, arXiv:1004.3398v1, 2010. 29

ER91. Eithan Ephrati and Jeffrey S. Rosenschein. The Clarke Tax as a Consensus Mechanism Among Automated Agents. In *Proceedings of the 9th AAAI Conference on Artificial Intelligence*, pages 173–178. AAAI Press, 1991. 1

ER97. Eithan Ephrati and Jeffrey S. Rosenschein. A Heuristic Technique for Multi-Agent Planning. *Annals of Mathematics and Artificial Intelligence*, 20(1–4):13–67, 1997. 1

ER10. Gábor Erdélyi and Jörg Rothe. Control Complexity in Fallback Voting. In *Proceedings of Computing: the 16th Australasian Theory Symposium*, Australian Computer Society Conferences in Research and Practice in Information Technology Series, pages 39–48, 2010. 29, 30

Fal08. Piotr Faliszewski. Nonuniform Bribery. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1569–1572. International Foundation for Autonomous Agents and Multiagent Systems, 2008. 26

Fel09. Michael R. Fellows. Towards Fully Multivariate Algorithmics: Some New Results and Directions in Parameter Ecology. In *Proceedings of the 20th International Workshop on Combinatorial Algorithms*, volume 5874 of *LNCS*, pages 2–10. Springer, 2009. 23, 38

FFL$^+$09. Henning Fernau, Fedor V. Fomin, Daniel Lokshtanov, Daniel Raible, Saket Saurabh, and Yngve Villanger. Kernel(s) for Problems with No Kernel: On Out-Trees with Many Leaves. In *Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science*, volume 3 of *LIPIcs*, pages 421–432. Schloss Dagstuhl, 2009. 34

FFL$^+$10. Henning Fernau, Fedor V. Fomin, Daniel Lokshtanov, Matthias Mnich, Geevarghese Philip, and Saket Saurabh. Ranking and Drawing in Subexponential Time. In *Proceedings of the 21st International Workshop on Combinatorial Algorithms*, volume 6460 of *LNCS*, pages 337–348. Springer, 2010. 16, 32

FG06. Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer Verlag, 2006. 2, 3, 4, 36

FHH09. Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. How Hard Is Bribery in Elections? *Journal of Artificial Intelligence Research*, 35:485–532, 2009. 26, 28

FHH10. Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. Using Complexity to Protect Elections. *Communications of the ACM*, 53(11):74–82, 2010. 1, 31

FHH11. Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. Multimode Control Attacks on Elections. *Journal of Artificial Intelligence Research*, 40:305–351, 2011. 28, 29

FHHR09a. Piotr Faliszewski, Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. A Richer Understanding of the Complexity of Election Systems. *Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz*, pages 375–406, 2009. 1

FHHR09b. Piotr Faliszewski, Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Llull and Copeland Voting Computationally Resist Bribery and Constructive Control. *Journal of Artificial Intelligence Research*, 35:275–341, 2009. 10, 26, 27, 28, 29, 30, 32, 34, 38, 39

Fis77. Peter C. Fishburn. Condorcet Social Choice Functions. *SIAM Journal on Applied Mathematics*, 33(3):469–489, 1977. 8, 10

FJL$^+$10. Michael R. Fellows, Bart Jansen, Daniel Lokshtanov, Frances A. Rosamond, and Saket Saurabh. Determining the Winner of a Dodgson Election is Hard. In *Proceedings of the 29th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 459–469. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010. 21, 33

FKS03. Ronald Fagin, Ravi Kumar, and Dandapani Sivakumar. Efficient Similarity Search and Classification via Rank Aggregation. In *Proceedings of the 22nd ACM SIGMOD International Conference on Management of Data*, pages 301–312. ACM, 2003. 1, 9

FS11. Lance Fortnow and Rahul Santhanam. Infeasibility of Instance Compression and Succinct PCPs for NP. *Journal of Computer and System Sciences*, 77(1):91–106, 2011. 33

Gae09. Wulf Gaertner. *A Primer in Social Choice Theory–LSE Perspectives in Economic Analysis*. Oxford University Press, revised edition, 2009. 6

Gib73. Allan Gibbard. Manipulation of Voting Schemes: A General Result. *Econometrica*, 41(4):587–601, 1973. 25

GMN09. Jiong Guo, Hannes Moser, and Rolf Niedermeier. Iterative Compression for Exactly Solving NP-Hard Minimization Problems. In *Algorithmics of Large and Complex Networks*, volume 5515 of *LNCS*, pages 65–80. Springer, 2009. 38

GN07. Jiong Guo and Rolf Niedermeier. Invitation to Data Reduction and Problem Kernelization. *ACM SIGACT News*, 38(1):31–45, 2007. 4, 32

Goo54. Leo A. Goodman. On Methods of Amalgamation. In R. M. Thrall, C. H. Coombs, and R. L. Davis, editors, *Decision Processes*, pages 39–48. John Wiley and Sons, Inc., 1954. 10

HHR97. Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Exact Analysis of Dodgson Elections: Lewis Caroll's 1876 Voting System is Complete for Parallel Access to NP. *Journal of the ACM*, 44(6):806–825, 1997. 9

HHR07. Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Anyone but Him: The Complexity of Precluding an Alternative. *Artificial Intelligence*, 171(5-6):255–285, 2007. 28, 29, 30

HHR09. Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Hybrid Elections Broaden Complexity-Theoretic Resistance to Control. *Mathematical Logic Quarterly*, 55(4):397–424, 2009. 29

HSV05. Edith Hemaspaandra, Holger Spakowski, and Jörg Vogel. The Complexity of Kemeny Elections. *Theoretical Computer Science*, 349(3):382–391, 2005. 9, 19

Hud04. Olivier Hudry. A Note On "Banks Winners in Tournaments Are Difficult to Recognize" by G. J. Woeginger. *Social Choice and Welfare*, 23(1):113–114, 2004. 21

IP01. Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *Journal of Computer and System Sciences*, 62:367–375, 2001. 4

IPZ01. Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. 4, 36

JSA08. Benjamin N. Jackson, Patrick S. Schnable, and Srinivas Aluru. Consensus Genetic Maps as Median Orders from Inconsistent Sources. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(2):161–171, 2008. 1, 9

Kem59. John G. Kemeny. Mathematics Without Numbers. *Daedalus*, 88:571–591, 1959. 8, 9

KL05. Kathrin Konczak and Jérôme Lang. Voting Procedures with Incomplete Preferences. In *Proceedings of IJCAI'05 Multidisciplinary Workshop on Advances in Preference Handling*, pages 124–129, 2005. 21, 24

KNU11. Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. Deconstructing Intractability—A Multivariate Complexity Analysis of Interval Constrained Coloring. *Journal of Discrete Algorithms*, 9:137–151, 2011. 38

KS10. Marek Karpinski and Warren Schudy. Faster Algorithms for Feedback Arc Set Tournament, Kemeny Rank Aggregation and Betweenness Tournament. In *Proceedings of the 21st International Symposium on Algorithms and Computation*, volume 6506 of *LNCS*, pages 3–14. Springer, 2010. 15, 16, 18, 32, 39

KT06. Jon Kleinberg and Éva Tardos. *Algorithm Design*. Addison Wesley, 2006. 15

LB11. Tyler Lu and Craig Boutilier. Budgeted Social Choice: From Consensus to Personalized Decision Making. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 280–286, 2011. 13

Len83. Hendrik W. Lenstra. Integer Programming with a Fixed Number of Variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. 21, 23, 30, 34, 38

Lev75. Arthur Levenglick. Fair and Reasonable Election Systems. *Behavioral Science*, 20(1):34–46, 1975. 9

LFZL09. Hong Liu, Haodi Feng, Daming Zhu, and Junfeng Luan. Parameterized Computational Complexity of Control Problems in Voting Systems. *Theoretical Computer Science*, 410:2746–2753, 2009. 30

Lok09. Daniel Lokshtanov. *New Methods in Parameterized Algorithms and Complexity*. PhD thesis, University of Bergen, 2009. 34

LR08. Claudia Lindner and Jörg Rothe. Fixed-Parameter Tractability and Parameterized Complexity Applied to Problems From Computational Social Choice. Supplement in the *Mathematical Programming Glossary*, October 2008. 2

LZ10. Hong Liu and Daming Zhu. Parameterized Complexity of Control Problems in Maximin Election. *Information Processing Letters*, 110(10):383–388, 2010. 30

MG06. Jiří Matoušek and Bernd Gärtner. *Understanding and Using Linear Programming (Universitext)*. Springer, 2006. 34

Mon95. Burt L. Monroe. Fully Proportional Representation. *American Political Science Review*, 89(4):925–940, 1995. 13

Mou91. Hervé Moulin. *Axioms of Cooperative Decision Making*. Cambridge University Press, 1991. 11, 12

MR99. Meena Mahajan and Venkatesh Raman. Parameterizing Above Guaranteed Values: MaxSat and MaxCut. *Journal of Algorithms*, 31(2):335–354, 1999. 18

MRS09. Meena Mahajan, Venkatesh Raman, and Somnath Sikdar. Parameterizing Above or Below Guaranteed Values. *Journal of Computer and System Sciences*, 75:137–153, 2009. 15, 18

MRS11. Neeldhara Misra, Venkatesh Raman, and Saket Saurabh. Lower Bounds on Kernelization. *Discrete Optimization*, 8(1):110–128, 2011. 33

Nie06. Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, February 2006. 2, 3, 4, 34, 36

Nie10. Rolf Niedermeier. Reflections on Multivariate Algorithmics and Problem Parameterization. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science*, volume 5 of *LIPIcs*, pages 17–32, 2010. 23, 38

Nur87. Hannu Nurmi. *Comparing Voting Systems*. Kluwer Academic Publishers, 1987. 6

Pap94. Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994. 2

PRVW11. Maria Silvia Pini, Francesca Rossi, K. Brent Venable, and Toby Walsh. Incompleteness and Incomparability in Preference Aggregation: Complexity Results. *Artificial Intelligence*, 175:1272–1289, 2011. 24, 38

PRZ08. Ariel D. Procaccia, Jeffrey S. Rosenschein, and Aviv Zohar. On the Complexity of Achieving Proportional Representation. *Social Choice and Welfare*, 30:353–362, 2008. 13

RBLR11. Jörg Rothe, Dorothea Baumeister, Claudia Lindner, and Irene Rothe. *Einführung in Computational Social Choice: Individuelle Strategien und kollektive Entscheidungen beim Spielen, Wählen und Teilen*. Spektrum Akademischer Verlag, 2011. 1, 6

RS07. Venkatesh Raman and Saket Saurabh. Improved Fixed Parameter Tractable Algorithms for Two "Edge" Problems: MAXCUT and MAXDAG. *Information Processing Letters*, 104(2):65–72, 2007. 15

RSV03. Jörg Rothe, Holger Spakowski, and Jörg Vogel. Exact Complexity of the Winner Problem for Young Elections. *Theory of Computing Systems*, 36(4):375–386, 2003. 10, 21

RSV04. Bruce A. Reed, Kaleigh Smith, and Adrian Vetta. Finding Odd Cycle Transversals. *Operations Research Letters*, 32:299–301, 2004. 38

Sat75. Mark Allen Satterthwaite. Strategy-Proofness and Arrow's Conditions: Existence and Correspondence Theorems for Voting Procedures and Social Welfare Functions. *Journal of Economic Theory*, pages 187–217, 1975. 25

Scu07. David W. Sculley. Rank Aggregation for Similar Items. In *Proceedings of the 7th SIAM International Conference on Data Mining*, pages 587–592, 2007. 9

SEF11. Ildikó Schlotter, Edith Elkind, and Piotr Faliszewski. Campaign Management under Approval-Driven Voting Rules. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pages 726–731. AAAI Press, 2011. 27

Sim09. Narges Simjour. Improved Parameterized Algorithms for the Kemeny Aggregation Problem. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation*, volume 5917 of *LNCS*, pages 312–323. Springer, 2009. 16, 18

Sla61. Patrick Slater. Inconsistencies in a Schedule of Paired Comparisons. *Biometrika*, 48(3–4):303–312, 1961. 21

Sni08. Moshe Sniedovich. Wald's Maximin Model: A Treasure in Disguise! *Journal of Risk Finance*, 9(3):287–291, 2008. 10

Tay05. Alan D. Taylor. *Social Choice and the Mathematics of Manipulation.* Cambridge University Press, 2005. 6

Wal49. Abraham Wald. Statistical Decision Functions. *The Annals of Mathematical Statistics*, 20(2), 1949. 8, 10

Wal07. Toby Walsh. Uncertainty in Preference Elicitation and Aggregation. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 3–8. AAAI Press, 2007. 23

Woe03. Gerhard J. Woeginger. Banks Winners in Tournaments are Difficult to Recognize. *Social Choice and Welfare*, 20(3):523–528, 2003. 21, 39

XC11. Lirong Xia and Vincent Conitzer. Determining Possible and Necessary Winners under Common Voting Rules Given Partial Orders. *Journal of Artificial Intelligence Research*, 41:25–67, 2011. 22, 23, 24, 30

YL78. H. P. Young and Arthur Levenglick. A Consistent Extension of Condorcet's Election Principle. *SIAM Journal on Applied Mathematics*, 35(2):285–300, 1978. 9

You77. H. P. Young. Extending Condorcet's Rule. *Journal of Economic Theory*, 16:335–353, 1977. 8, 9, 10, 21