A Multivariate Complexity Analysis of Voting Problems

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium (Dr. rer. nat.)

vorgelegt dem Rat der Fakultät für Mathematik und Informatik
der Friedrich-Schiller-Universität Jena

von Dipl.-Bioinf. Nadja Betzler
geboren am 08. 08. 1979 in Stuttgart

Gutachter:

- Prof. Dr. Rolf Niedermeier (Friedrich-Schiller-Universität Jena / Technische Universität Berlin)
- Prof. Dr. Lane A. Hemaspaandra (University of Rochester)
- Prof. Dr. Jörg Rothe (Heinrich-Heine-Universität Düsseldorf)

Tag der öffentlichen Verteidigung: 19. November 2010

Zusammenfassung

Die Arbeit beschäftigt sich mit einer multivariaten Komplexitätsanalyse kombinatorischer Probleme im Wahlkontext. Wahlen beschreiben hierbei den Prozess einer gemeinsamen Entscheidungsfindung mehrerer Parteien. Formal betrachtet besteht eine Wahl aus einer Multimenge von Stimmabgaben über einer Menge von Kandidaten (oder Alternativen). In dem üblichen Szenario liegen die Stimmabgaben hierbei als Präferenzlisten vor, das heißt als lineare Ordnung der Kandidaten. Neben der offensichtlichen Anwendung bei politischen Wahlen treten informatikrelevante „Wahlprobleme“ beispielweise in der Bioinformatik, im Kontext von Datenbanken und Suchmaschinen im Internet auf.

Die auftretenden Probleme reichen hierbei von der Berechnung eines Gewinners anhand eines bestimmten Wahlverfahrens über strategisches Wählen bis hin zur Beeinflussung des Wahlausgangs durch externe Agenten. Hinzu kommen verschiedene Szenarien basierend auf unterschiedlicher Verfügbarkeit von Information. Viele dieser Probleme sind NP-hart. Wir untersuchen den Ansatz einer multivariaten Komplexitätsanalyse mit besonderem Schwerpunkt auf Parametrisierter Komplexität als möglichen Ausweg. Eine parametrisierte Komplexitätsanalyse basiert auf einer zweidimensionalen Sichtweise. Zusätzlich zu der Eingabegröße betrachtet man einen Parameter, beispielsweise die Lösungsgröße oder die Anzahl der Kandidaten. Ein Problem ist „*fixed-parameter tractable*“, wenn es einen Algorithmus gibt, dessen nichtpolynomieller Anteil der Laufzeit nur von einer Funktion des Parameters abhängt. Wenn der betrachtete Parameter klein gemessen an der Eingabegröße ist, kann dies zu effizienten Algorithmen führen. Im Wahlkontext gibt es beispielsweise Situationen mit vielen Wählern aber nur wenigen Kandidaten, z.B. politische Wahlen. In solchen Fällen kann die Beschränkung der inhärenten kombinatorischen Explosion von NP-harten Problemen auf eine Funktion, die nur von der Anzahl der Kandidaten abhängt, zu effizienten Algorithmen führen. Eine multivariate Analyse erweitert das Konzept der parametrisierten Komplexität, so dass mehrere Parameter auf einmal untersucht werden.

Obwohl die betrachteten Probleme meist eine Vielzahl von Parametrisierungen zulassen, die sinnvolle Szenarien abdecken, gibt es bisher nur sehr wenige Studien in diese Richtung. Das Ziel dieser Arbeit ist daher, die Untersuchung der multivariaten und insbesondere der parametrisierten Komplexität für wichtige Wahlprobleme voranzutreiben. Wir betrachten drei Arten von Problemen, die zu einer Einteilung der Arbeit in drei Teile führen:

- Die Berechnung eines Gewinners unter vollständiger Information.
- Die Berechnung eines möglichen Gewinners ausgehend von unvollständiger Information.
- Die Beeinflussung eines Wahlausgangs durch einen externen Agenten mittels Hinzufügens oder Löschens von Kandidaten.

Die einzelnen Ergebnisse werden im Folgenden zusammengefasst.

Kapitel 1 und 2 führen in die Thematik von algorithmischen Wahlproblemen ein. Kapitel 1 enthält eine kurze Zusammenfassung der Ergebnisse und führt relevante Notation und Konzepte ein. Kapitel 2 gibt einen Überblick über für die Arbeit wesentlichen Konzepte aus dem Bereich der Wahlforschung. Im Anschluss werden bisherige Arbeiten der theoretischen Informatik und künstlichen Intelligenz, die sich mit Wahlsystemen beschäftigen, diskutiert. Hierbei wird besonderer Wert auf die Darstellung der bisher bekannten Ergebnisse im Bereich der parametrisierten Komplexität gelegt.

Teil 1 (Kapitel 3,4,5 und 6) Für einige Wahlsysteme ist es bereits NP-hart einen Gewinner zu berechnen. In diesem Teil der Arbeit untersuchen wir die Parametrisierte Komplexität bezüglich drei solcher Wahlsysteme. Unser Hauptaugenmerk liegt hierbei auf dem Rank Aggregation oder Kemeny Score Problem. Eine prominente Anwendung ergibt sich beispielweise im Kontext von Metasuchmaschinen, wobei Eingabelisten von verschiedenen Suchmaschinen zusammenfasst werden müssen. Formal kann man das zugehörige Problem wie folgt definieren. Gegeben ist eine Multimenge von Präferenzlisten und diese sollen zu einer Konsensliste zusammengefasst werden, so dass die Summe der Abstände der Konsensliste zu den einzelnen Eingabelisten minimiert wird. Der Abstand bezeichnet hierbei die Anzahl der Inversionen, dass heißt die Anzahl der unterschiedlich angeordneten Kandidatenpaare.

In Kapitel 3 untersuchen wir die parametrisierte Komplexität von Rank Aggregation bezüglich verschiedener Parametrisierungen. Insbesondere identifizieren wir hierbei Parameter, die strukturelle Eigenschaften messen, zum Beispiel den „durchschnittlichen Abstand der Eingabelisten“ d_a oder den „Positionsbereich in dem ein Kandidat auftritt“. Ein kleiner durchschnittlicher Abstand scheint in Anwendungen plausibel in denen man davon ausgehen kann, dass es eine „beste“ Konsensliste gibt, die einzelnen Wähler diese aber nur verrauscht wiedergeben, wie in dem obigen Beispiel die Suchmaschinen. Unsere Ergebnisse beinhalten mehrere dynamische Programmieralgorithmen, die „fixed-parameter tractability“ bezüglich der Parameter „Anzahl Kandidaten“, d_a und „maximaler Positionsbereich eines Kandidaten“ zeigen.

In Kapitel 4 erweitern wir die algorithmischen Ergebnisse aus dem vorherigen Kapitel indem wir Datenreduktionsregeln angeben, die in Polynomzeit ausführbar sind. Zur Analyse benutzen wir die neueingeführte Analysetechnik des „Partiellen Problemkerns“, die zu beweisbaren „fixed-parameter tractability“-Resultaten führt. Ein wesentlicher Punkt ist dabei die Einführung eines neuen Parameters, der die Anzahl der „Konfliktpaare“ der Kandidatenpaare bezüglich verschiedener Mehrheiten misst.

Kapitel 5 beschreibt eine experimentelle Evaluierung der Datenreduktionsregeln sowie einiger Algorithmen aus Kapitel 3 anhand von Daten aus Sportwettkämpfen und Suchmaschinen. Hierbei zeigen wir, dass die Datenreduktionsregeln das Berechnen von Konsenslisten für Instanzen erlaubt, die vorher nicht lösbar waren.

Kapitel 6 beschäftigt sich mit der Gewinnerbestimmung für zwei Wahlsysteme, die von Dodgson und Young eingeführt wurden. In beiden dieser Wahlsysteme beschreibt die Punktzahl eines Kandidaten die Anzahl der „Operationen“, die nötig sind, um diesen zu einem Condorcet-Gewinner zu machen, das heißt zu einem Kandidaten, der alle anderen im direkten paarweisen Vergleich schlägt. Die Operation für Dodgson ist das Vertauschen benachbarter Kandidaten in einer Stimmabgabe und bei Young das Löschen von Stimmabgaben. Unser Hauptergebnis ist, dass die Berechnung der Punktzahl eines Kandidaten bezüglich der Anzahl der Operationen für Dodgson „fixed-parameter tractable“ ist, hingegen das analoge Problem für Young W[2]-vollständig ist.

Teil 2: Berechnung eines möglichen Gewinners (Kapitel 7,8,9 und 10) In dem Standardmodell einer Wahl geben die Wähler ihre Stimmabgaben als lineare Ordnung über alle Kandidaten ab. In vielen Situation scheint dies unrealistisch zu sein oder man möchte schon etwas über den *möglichen* Ausgang einer Wahl erfahren, bevor die gesamte Information vorliegt. Dies führt direkt zu dem Possible Winner Problem, das fragt, ob eine vorliegenden Multimenge partieller Ordnungen zu einer Menge von linearen Ordnungen erweitert werden kann, so dass ein ausgewiesener Kandidat gewinnt. In diesem Teil der Arbeit betrachten wir das Possible Winner Problem für eine Menge von Wahlsystemen, in denen jeder Kandidat Punkte abhängig von seiner Position in den Stimmabgaben bekommt. Beispiele für solche Systeme sind das einfache Mehrheitswahlrecht (ein Kandidat bekommt einen Punkt für jede Stimmabgabe, in der er vorne steht), sowie Borda, in dem für m Kandidaten, der erste Kandidat einer Liste m Punkte, der zweite $m-1$ Punkte, usw. bekommt.

In Kapitel 7 betrachten wir die Schwierigkeit des Possible Winner Problems abhängig von dem betrachteten punktebasiertem Wahlsystem. Auch unter Ausnutzung einiger Ergebnisse aus der Literatur geben wir eine komplette Dichotomie an: Das Possible Winner Problem kann für einfache Mehrheitswahlen und sogenannte Vetowahlen in Polynomzeit gelöst werden und ist NP-vollständig für alle anderen Fälle.

Kapitel 8 beschäftigt sich mit einer parametrisierten Komplexitätsanalyse als möglichen Ausweg aus der im vorherigen Kapitel gezeigten NP-Vollständigkeit. Neben der Betrachtung von Parametern, die den Grad der Unvollständigkeit einer Eingabeinstanz messen zeigen wir, dass das Possible Winner Problem fixed-parameter tractable bezüglich der Anzahl der Kandidaten ist. Außerdem zeigen wir für Borda, sowie für k-Approval (die besten k Kandidaten einer Stimmabgabe bekommen einen Punkt), dass das Possible Winner Problem schon für eine konstante Anzahl von Wählern NP-vollständig ist.

Kapitel 9 zeigt fixed-parameter tractability bezüglich zweier „kombinierter“ Parameter für Possible Winner unter dem k-Approval System. Die Resultate umfassen die Entwicklung von Problemkernen sowie einen Beweis der Nichtexistenz eines Problemkerns polynomieller Größe (unter einigen komplexitätstheoretischen Annahmen).

Kapitel 10 schließt Teil 2 ab und beinhaltet einige Fragestellungen für zukünftige Arbeiten.

Teil 3: Wahlkontrolle (Kapitel 11) Der letzte und kürzeste Teil dieser Arbeit beschäftigt sich mit der Beeinflussung des Wahlausgangs durch externe Agenten. Wir betrachten das Problem, ob ein ausgewiesener Kandidat durch das Löschen oder Hinzufügen von anderen Kandidaten zu einem Gewinner gemacht werden kann, oder ob

verhindert werden kann, dass dieser gewinnt. Wir erweitern die bisherigen Untersuchungen für dieses Problem indem wir die parametrisierte Komplexität für die Fälle untersuchen, dass nur eine begrenzte Anzahl von Kandidaten gelöscht bzw. hinzugefügt werden dürfen.

Kapitel 12 umfasst eine kurze Zusammenfassung der Ergebnisse dieser Arbeit sowie mögliche Fragestellungen für weitere Forschungsthemen im Bereich der multivariaten Komplexitätsanalyse von Wahlproblemen.

Preface

This thesis summarizes my results on a multivariate computational complexity analysis for some voting problems. My research was funded by the Deutsche Forschungsgemeinschaft (DFG) since November 2006 under various projects (PEAL (NI 369/1), PIAF (NI 369/4), DARE (GU 1023/1-1), and PAWS (NI 369/10)). I owe sincere thanks to this support and to Rolf Niedermeier and Jiong Guo who initiated these research projects. In particular, I am very grateful to my supervisor Rolf Niedermeier for giving me the opportunity to work in his group and for his support in my research and the development of this thesis. I also thank my (partially former) colleagues René van Bevern, Robert Bredereck, Michael Dom, Sepp Hartung, Falk Hüffner, Jiong Guo, Christian Komusiewicz, Johannes Uhlmann, Jörg Vogel, and Mathias Weller for providing a pleasant and stimulating working atmosphere. Many thanks also to my coauthors Yoram Bachrach, René van Bevern, Robert Bredereck, Britta Dorn, Piotr Faliszewski, Michael R. Fellows, Susanne Hemmann, Falk Hüffner, Christian Komusiewicz, Rolf Niedermeier, Frances A. Rosamond, and Johannes Uhlmann for fruitful cooperations and to Robert Bredereck for his great implementation and experimentation work in the Kemeny project. Moreover, I would like to thank the anonymous referees from various conferences and journals who helped to improve the quality of this work. Last but not least I am grateful to the organizers of several Dagstuhl seminars for their invitations.

Regarding the design of the thesis cover, I am very grateful to Britta Dorn for providing the drawing of a joint decision making process. Any resemblance to real persons is purely coincidental.

This thesis emerges from collaborations with various research partners. In the following, I describe my specific contributions. Moreover, I provide an overview of the publications establishing the basis for this thesis. In addition, during my time at FSU Jena, I contributed to the publications [7, 17, 23, 31, 135], which are not part of this thesis.

Part I: Winner Determination. Part I of the thesis is concerned with the determination of a winner for the three voting systems due to Kemeny (Chapters 3, 4, and 5), Dodgson, and Young (Chapter 6).

A first systematic analysis as described in Chapter 3 has been obtained in coop-

eration with Jiong Guo, Michael R. Fellows, Rolf Niedermeier and Frances A. Rosamond. Some of the results were obtained in discussions with all collaborators. My contributions can be summarized as follows. I came up with a first fixed-parameter algorithm with respect to the "maximum KT-distance $d_{\max}$ between two input votes" as published in the conference paper [20], which appeared in the proceedings of the *4th International Conference on Algorithmic Aspects in Information and Management (AAIM'08)*. The corresponding dynamic programming algorithm had a running time of $O^*(d_{\max}!)$. This result was further improved by Jiong Guo and me as follows. Jiong provided an important observation that made the dynamic programming applicable to the "stronger" parameter "average KT-distance" d_a and, based on this, I provided a refined dynamic programming procedure improving the exponential part of the running time from $O^*(d_a!)$ to $O^*(16^{d_a})$. In addition, I showed that the same dynamic programming procedure can be used to work for the parameter "maximum candidate range" and provided an NP-hardness proof for constant "average candidate range". The refined dynamic programming algorithm appears in the proceedings of the *8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'09)* [22]. The journal version [21], which appeared in *Theoretical Computer Science*, combines the results from [20] and [22]. The conference paper [22] was also presented at the *2nd International Workshop on Computational Social Choice (COMSOC'08)*.

Chapter 4 exploits the concept of partial kernelization for Kemeny Score. The basic idea of partial kernelization goes back to Jiong Guo and has been introduced for several median problems. Herein, the detailed proofs leading to partial kernels of quadratic size with respect to the "average KT-distance" for Kemeny Score (with Ties) were worked out by Jiong Guo and me. The general framework and concept of partial kernelization was shaped in discussions with Jiong Guo, Christian Komusiewicz, and Rolf Niedermeier. The results for several median problems were published in the proceedings of the *9th Latin American Theoretical Informatics Symposium (LATIN'10)* [24] and the full version is to appear in *Journal of Computer and System Sciences* [25]. Chapter 4 combines parts of the results from the paper [25] with several new results described in the following. In close cooperation with Robert Bredereck, I obtained an improvement of the partial kernel size from quadratic to linear and showed some "tightness" results for corresponding data reduction rules. These results are accepted for publication at the *5th International Symposium on Parameterized and Exact Computation (IPEC'10)* and were also presented at the *3rd International Workshop on Computational Social Choice (COMSOC'10)* [16]. Chapter 4 contains the theoretical results from [16] while the experimental results from [16] are discussed in Chapter 5.

Chapter 5 provides an experimental evaluation of some of our fixed-parameter algorithms for Kemeny Score with particular focus on data reduction rules [16]. Besides evaluating our algorithms for real-world data, we designed a publicly available software package. The implementation work was accomplished by Robert Bredereck working as a research student mainly under my supervision.

Chapter 6 is concerned with the parameterized complexity of Dodgson and Young elections. Inspired by some pointers to the literature provided by Jörg Vogel, the project was initiated by Jiong Guo, Rolf Niedermeier, and me. I contributed a dynamic programming algorithm leading to fixed-parameter tractability for Dodgson

Score and showed the W[2]-hardness of Young Score, in both cases with respect to the corresponding score parameter. Jiong Guo further settled the parameterized complexity of Young Score by providing W[2]-membership (for which the proof is not contained in this thesis). The results of Chapter 6 were presented at the *11th Scandinavian Workshop on Algorithm Theory (SWAT'08)* [26] and appeared in *Information and Computation* [27].

Part II: Possible Winner Determination. Part II investigates the problem of finding possible winners under scoring rules. I initiated the research on this topic and came up with all major results.

Chapter 7 aims at a full classification of the computational complexity of Possible Winner for scoring rules. I provided the overall framework as well as the ideas for the many-one reductions. I am very grateful to Britta Dorn for her help with writing and working out several details of this voluminous work. Large parts of this chapter follow the journal paper [19], which appeared in *Journal of Computer and System Sciences*, for which the conference version [18] was presented at the *34th International Symposium on Mathematical Foundations of Computer Science (MFCS'09)*.

Chapter 8 is concerned with a parameterized complexity analysis of Possible Winner for scoring rules with respect to several single parameters. In the course of this project, Susanne Hemmann prepared her diploma thesis under Rolf Niedermeier's and my supervision. I had the main ideas for all results in this chapter and also worked out the details as provided in this chapter. Most of the results appeared in the proceedings of the *21st International Joint Conference on Artificial Intelligence (IJCAI-09)* [28].

Chapter 9 extends the range of fixed-parameter algorithms for Possible Winner for k-approval voting by providing kernelization results and showing (presumably) nonexistence of a polynomial kernel for combined parameters. All results were obtained by me. The results appeared in the proceedings of the *35th International Symposium on Mathematical Foundations of Computer Science (MFCS'10)* [15].

Part III: Candidate Control. The last part of the thesis consists of Chapter 11 devising a systematic parameterized complexity analysis of candidate control in Copeland elections and some closely related digraph problems. I initiated the research on this topic and provided the main part of the results after fruitful discussions with Johannes Uhlmann. The paper was presented at the *2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA)* [29] and a journal version appeared in *Theoretical Computer Science* [30]. Chapter 11 basically follows [30].

Contents

1 Introduction **1**
1.1 Basic definitions 4
1.1.1 Elections and voting systems 4
1.1.2 Graphs and digraphs 5
1.2 Computational Complexity 5
1.2.1 P versus NP 5
1.2.2 Parameterized complexity 6
1.2.3 Multivariate algorithmics 7
1.3 Algorithm design tools 7
1.3.1 Kernelization 8
1.3.2 Size-bounded search trees 8
1.3.3 Integer linear programming 8

2 Problems and results in context **11**
2.1 A brief introduction to voting 11
2.1.1 Voting rules and winner determination 12
2.1.2 Strategic voting and possible winners 14
2.1.3 Influences by external agents 14
2.2 Computational aspects of voting 15
2.2.1 Winner determination 16
2.2.2 Manipulation and possible winners 16
2.2.3 Control 17
2.3 Multivariate algorithmic results 17

I Winner Determination **21**

3 Parameterizations for Kemeny **23**
3.1 Introduction 23
3.2 Parameterizations and our results. 25
3.2.1 Range versus distance parameterizations 27
3.3 Parameterization by the number of candidates 29
3.4 Parameterization by the Kemeny score 29

3.5 Parameterization by the average KT-distance . . . 31
3.5.1 A crucial observation . . . 32
3.5.2 Dynamic programming algorithm . . . 34
3.6 Parameterizations by the candidate range . . . 38
3.6.1 Maximum range . . . 38
3.6.2 Average range . . . 39
3.7 Ties and incomplete votes . . . 40
3.7.1 Kemeny Score with Ties . . . 40
3.7.2 Kemeny Score with Incomplete Votes . . . 41
3.8 Further fixed-parameter algorithms . . . 41
3.9 Conclusion . . . 43

4 Partial kernelization for Kemeny 45
4.1 Framework and basic definitions . . . 46
4.2 Dirtiness and a polynomial-time special case . . . 47
4.3 Data reduction rules and partial kernelization . . . 49
4.3.1 Exploiting $\geq_{3/4}$-majorities . . . 50
4.3.2 Tightness of the 3/4-Majority Rule . . . 51
4.3.3 Exploiting $>_{2/3}$-majorities . . . 54
4.4 Conclusion . . . 56

5 Experimental results for Kemeny 59
5.1 Implemented algorithms . . . 60
5.1.1 Data reduction rules . . . 60
5.1.2 Exact solution algorithms . . . 61
5.2 Experimental results . . . 62
5.2.1 Sport competitions . . . 62
5.2.2 Search result rankings . . . 63
5.2.3 Impact rankings . . . 66
5.3 Conclusion . . . 67

6 Dodgson and Young voting 69
6.1 Dodgson Score . . . 72
6.1.1 Dynamic programming algorithm . . . 72
6.1.2 Allowing ties . . . 77
6.2 Young Score . . . 78
6.3 Conclusion . . . 82

II Possible Winner Determination 85

7 A dichotomy for pure scoring rules 87
7.1 Motivation and known results . . . 87
7.2 Definitions . . . 89
7.3 General strategy . . . 91
7.3.1 A general scheme to construct linear votes . . . 92
7.4 Plurality and veto . . . 94
7.5 An unbounded number of positions with different score values . . . 95

7.6 An unbounded number of positions with equal score values 99
7.6.1 An unbounded number of equal score values and $\alpha_2 \neq \alpha_{m-1}$. 99
7.6.2 Scoring vectors with $\alpha_1 > \alpha_2 = \cdots = \alpha_{m-1} > 0$ 112
7.7 Putting all together 121
7.8 Conclusion 123

8 A parameterized complexity study for scoring rules 125
8.1 Number of candidates 127
8.2 Number of votes 128
8.2.1 k-approval 129
8.2.2 Borda 132
8.3 Measures of incompleteness 137
8.3.1 Maximum number of undetermined pairs per vote 137
8.3.2 Total number of undetermined pairs 139
8.3.3 Further parameters measuring incompleteness 143

9 Combined parameters for k-approval voting 145
9.1 Fixed number of zero-positions 147
9.1.1 Problem kernel 147
9.1.2 Parameterized algorithms 149
9.2 Fixed number of one-positions 152
9.2.1 Problem kernels 152
9.2.2 Kernel lower bound 158
9.3 Conclusion 163

10 Conclusion Part II 165

III Candidate Control 171

11 Candidate control for Copeland and plurality 173
11.1 Candidate control and related digraph problems 175
11.1.1 Control problems 175
11.1.2 Digraph problems 175
11.1.3 Contributions and organization 177
11.2 Parameterized complexity of the digraph problems 178
11.2.1 Vertex deletion 179
11.2.2 Vertex addition 184
11.2.3 W[2]-membership 187
11.3 Parameterized complexity of candidate control 188
11.3.1 Llull and Copeland voting 189
11.3.2 Number of votes as parameter 189
11.3.3 Number of deleted/added candidates as parameter 193
11.4 Conclusion 197

12 Summary and future research directions **199**
12.1 Summary of results 199
12.2 Future challenges 200
12.2.1 Problem-oriented approaches 200
12.2.2 Technique-oriented approaches 201
12.2.3 Parameter-oriented approaches 202

Chapter 1

Introduction

Voting asks to reach a joint decision based on the preferences of multiple parties. At a first sight, classical scenarios such as political elections may come into one's mind. However, at a second sight, voting seems to be omnipresent in our lives: Voting scenarios rise from mundane situations like making a joint decision on the holiday destination or a restaurant to the decision about job applicants or awarding honors to multiagent settings. In particular, over the last decades computer science itself increased the "need" for joint decision making; for example, deciding which job is the first to run on a machine, or aggregating the results from several search engines.

The study of the computational complexity of "voting problems" is an active area of research [53, 58, 96, 100, 103]. Important problems comprise winner determination according to certain voting protocols with desirable properties or manipulation. Unfortunately, many voting problems have turned out to be NP-hard. Furthermore, for many settings approximate solutions to such problems may be of limited interest. Clearly, in political elections nobody would be satisfied with an approximate winner determination, and similar observations hold for other applications. Hence, *exact* solutions are of particular relevance in this context. Given the NP-hardness of the problems, however, it seems inevitable to live with exponential-time algorithms. A way out of this dilemma can be provided by multivariate algorithmics which investigates whether specific relevant settings allow for efficient algorithms despite general NP-hardness results.

Voting comes with a large variety of different settings. This includes political elections with many voters and few candidates, contrasting a small committee selecting a winner out of a large set of candidates. Although this invites for studies that explore voting problems in the context of such specific settings, so far there are only few publications in this direction. This work aims at a multivariate approach capturing questions such as whether a problem becomes "easy" in case that there are only few voters or candidates, respectively.

An important ingredient of a multivariate analysis is the investigation of the parameterized complexity of NP-hard problems as pioneered by Downey and Fellows [76, 113, 171]. This theory is based on the concept of *fixed-parameter* algorithms, that is, exact algorithms that confine the combinatorial explosion of the running time

to a function depending *solely* on a considered parameter such as the number of votes or the number of candidates. Parameterized complexity allows for a more fine-grained analysis than a "classical analysis" that only distinguishes between NP-hardness and polynomial-time solvability for *constant* parameter values: For a problem instance of size n and a parameter k, a fixed-parameter algorithm runs in $f(k) \cdot n^{O(1)}$ time where f denotes a computable function. In contrast, a running time of $O(n^k)$ does not imply fixed-parameter tractability. In this sense, we advocate parameterized algorithmics as a helpful tool for better understanding and exploiting the numerous natural parameters occurring in voting scenarios with associated NP-hard combinatorial problems.

As mentioned above, the "nature" of voting problems invites for several parameterizations. All considered "voting problems" are defined on *elections* consisting of a multiset of preference lists (or votes) over a set of candidates and on a voting rule that selects a winner of this election. This directly leads to the two *standard voting parameters* "number of candidates" and "number of votes". In addition, we will identify *problem-specific parameterizations* capturing realistic scenarios.

In the following, we describe the organization of this work. In Chapter 2, we provide an introduction to voting problems and discuss related work. In particular, we give an overview of corresponding multivariate complexity results. Chapter 2 is followed by our results on a multivariate analysis of the computational complexity for three types of voting problems dividing the thesis into three parts. First, we focus on the basic problem of winner determination. Second, we investigate the more general setting of possible winner determination under incomplete information. Finally, we consider how an external agent can influence the outcome of an election in favor or disfavor of a candidate by adding or deleting further candidates. We now describe the parts and their individual chapters.

Part I: Winner determination (Chapters 3, 4, 5, and 6). For some voting systems the determination of a winner is NP-hard. We provide a multivariate complexity analysis for three such voting systems. The main focus lies on the Rank Aggregation or Kemeny Score problem (Chapters 3–5). Here one is given a multiset of preference lists and the goal is to find a consensus list that minimizes the sum of distances from the input list according to a natural distance measure. Rank Aggregation has numerous applications [77, 92, 138, 187], for example, the preference lists might correspond to several search engines and the goal of a meta-search engine is to combine them into one list. Moreover, "Kemeny elections" also fulfill desirable properties from the social choice point of view [201]. Our results from the first chapter of this part comprise a systematic analysis with respect to the parameterizations

- number of candidates,
- average distance between the input votes,
- average and maximum range of a candidate in the input votes, and
- the solution value Kemeny score.

Besides obtaining algorithmic and hardness results with respect to specific parameters we also provide a comparison of the parameter values and identify realistic scenarios motivating the individual parameterizations. Hence, this study may also be considered

as an introduction into the "art of parameterization". Chapter 4 further extends the algorithmic results for Kemeny Score by providing data reduction rules with provable performance guarantee leading to the concept of partial kernels. Chapter 5 is concerned with an experimental evaluation of the data reduction rules and some of the fixed-parameter algorithms providing promising results for real-world data.

The last chapter of this part, Chapter 6, investigates Dodgson and Young elections. In both voting systems the score of a candidate measures the distance from being a Condorcet winner, that is, a candidate beating every other candidate in pairwise comparison. The distance is measured by the editing operations "swapping neighboring candidates" (Dodgson) and "deleting votes" (Young). Our main result is that the corresponding problem Dodgson Score is fixed-parameter tractable with respect to the "number of editing operations" while Young Score is W[2]-complete and hence presumably not fixed-parameter tractable.

Part II: Possible Winner Determination (Chapters 7, 8, 9, and 10). In this part, we focus on possible winner determination for so-called "positional scoring rules". Basically, a positional scoring rule assigns a number of points to every candidate depending on its position in a vote. Famous examples comprise plurality where, per vote, the first candidate gets one point and all other candidates get zero points, and Borda where, for m candidates, within one vote the first candidate gets $m-1$ points, the second candidate gets $m-2$ points, and so on. The winner determination under "complete information" can be easily accomplished in polynomial time by adding the points of every candidate over all votes. However, in many realistic settings the voters may only provide incomplete information, that is, partial orders instead of "full" preference lists. This directly leads to the question whether a multiset of "partial orders" can be extended to a corresponding multiset of linear orders such that a specific candidate wins. As we will show in Chapter 7, except for the two simple scoring rules plurality and veto, the corresponding Possible Winner problem is NP-complete for all natural scoring rules. Herein, one remaining open case missing for a full dichotomy has been settled by Baumeister and Rothe [14].

The NP-completeness results for Possible Winner under almost all naturally appearing scoring rules motivate a multivariate analysis of the problem as provided in Chapter 8. We show fixed-parameter tractability with respect to the "number of candidates" for all scoring rules and provide NP-completeness results for a constant number of votes for the two important scoring rules Borda and k-approval. In addition, we investigate parameterizations measuring the "amount of incompleteness" of an instance.

In Chapter 9 we focus on k-approval voting where the first k candidates get one point each and the remaining candidates get zero points per vote. For Possible Winner under k-approval voting, we show fixed-parameter tractability with respect to combined parameters capturing realistic scenarios. Our results comprise polynomial-size problem kernels as well as nonexistence proofs of polynomial-size problem kernels and are among the first (non poly-) kernelization results for voting problems.

Chapter 10 concludes Part II and presents several tasks for future research.

Part III: Electoral control (Chapter 11). In electoral control, an external agent seeks to influence an election to reach certain goals. We focus on the problem whether

an agent can add or delete candidates such that a distinguished candidate becomes a winner or is prevented from winning. On the one hand, in this context computational hardness is considered as a desirable property [13] since in many settings control is clearly bad. On the other hand, there are legal scenarios such as persuading additional players to participate in a sport competition (like chess competitions in which usually every player plays against every other player) in which the external agent is interested in having an efficient strategy to reach her or his goal. We investigate the parameterized complexity of control of Copeland$^\alpha$ and plurality voting, two important and commonly used voting systems with respect to several parameterizations. An important part of this analysis is the identification of some closely related problems on directed graphs and corresponding studies of their computational complexities.

Chapter 12 summarizes the results of the thesis and provides some directions for future research distinguishing between problem-oriented, technique-oriented, and parameter-oriented approaches.

The remaining part of this introductory chapter provides some basic definitions and concepts of computational complexity and algorithm design.

1.1 Basic definitions

We introduce some basic concepts and definitions needed in several parts of the thesis. Definitions only relevant for one part or chapter will be provided there.

1.1.1 Elections and voting systems

An *election* (V, C) consists of a multiset V of n votes and a set C of m candidates (or alternatives). A *vote* (or *preference list*) is a linear order (i.e., a transitive, antisymmetric, and total relation) on C. For example, for $C = \{a, b, c\}$, the vote $a > b > c$ means that a is the best-liked and c the least-liked candidate in this vote. At some places, we specify a subset $D \subseteq C$ of candidates and the corresponding "reverse" subset $\overline{D}$ within a vote. This is to read as follows. Fix the candidates of C at an arbitrary order, then $\ldots > D > \ldots$ means that the candidates from D appear according to this fixed order and $\ldots > \overline{D} > \ldots$ means that the candidates from D appear reverse to the fixed order.

A *voting rule*[1] r is a function that maps an election to a subset a 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 *cowinners*.

A widely used voting rule is *plurality* where every voter can vote for one candidate and the candidates with the highest number of votes win. Plurality is a so-called "positional scoring rule" since the score of a candidate depends only on its position in every vote, that is, getting one point if it is ranked first and zero points at every other position. We also consider voting systems based on pairwise comparisons of candidates. To this end, we say a candidate beats another candidate in their *pairwise head-to-contest* if it is better positioned than the other candidate in more than half of the votes. A candidate beating every other candidate in the pairwise head-to-head

[1] Also called *voting correspondence*, *voting/election system*, or *(social) choice procedure*.

contest is a *Condorcet winner* [69]. A Condorcet winner does not always exist and an election has at most one Condorcet winner [69]. A voting system that elects a Condorcet winner if one exists fulfills the *Condorcet property*, which is desirable for many applications.

1.1.2 Graphs and digraphs

Several of our results make use of graphs or digraphs. We briefly introduce some relevant notations (see also [9, 70, 139] for basic definitions). For an undirected graph $G = (U, E)$ and a vertex $u \in U$, the *open neighborhood* $N(u)$ of u is the set of vertices adjacent to u. Moreover, $N[u] := N(u) \cup \{u\}$ is called the *closed neighborhood* of u. For a directed graph (digraph) $D = (W, A)$ and for a vertex $w \in W$, the set of *in-neighbors* of w is defined as $N_{\text{in}}(w) := \{u \in W \mid (u, w) \in A\}$ and the set of *out-neighbors* of w is given by $N_{\text{out}}(w) := \{u \in W \mid (w, u) \in A\}$. The *indegree* of w is defined as $d_{\text{in}}(w) := |N_{\text{in}}(w)|$ and the *outdegree* of w is defined as $d_{\text{out}}(w) := |N_{\text{out}}(w)|$. The *degree* of w is defined as $\deg(w) := d_{\text{in}}(w) + d_{\text{out}}(w)$. For a set of vertices $W' \subseteq W$, the *induced subgraph* $D[W']$ is the graph over the vertex set W' with arc set $\{(w, u) \in A \mid w, u \in W'\}$. In digraphs, we do not allow bidirected arcs and loops. An l-arc coloring $\mathcal{C} : A \to \{1, 2, \ldots, l\}$ is called *proper* if any two distinct arcs of the same color do not share a common vertex. A *tournament* is a digraph where, for every pair of vertices u and v, there is either (u, v) or (v, u) in the arc set.

For an undirected bipartite graph $(G \cup H, E)$ with vertex set $G \cup H$ and edge set $E \subseteq \{\{g, h\} \mid g \in G \text{ and } h \in H\}$, a *matching* denotes a subset $M \subseteq E$ such that for all $e, e' \in M$, $e \cap e' = \emptyset$. A vertex contained in e for an $e \in M$ is called *matching vertex* and, for $\{g, h\} \in M$, g and h are *matching neighbors*. A *maximum matching* is a matching with maximum cardinality.

1.2 Computational Complexity

The computational complexity of a problem can be measured by the resources needed to solve it (see [118, 176] for more details). In general, computational complexity theory aims at a classification of problems into respective "complexity classes". In the following, we briefly introduce the two complexity classes P and NP from classical complexity theory and give a short introduction in "parameterized complexity classes". This is followed by introducing the term of "multivariate algorithmics", a concept generalizing parameterized algorithmics.

In most places of the work we deal with *decision problems* but all algorithmic results can be modified easily to construct a corresponding "solution". Formally, a decision problem is encoded as language $L \subseteq \Sigma^*$ over an alphabet Σ.

1.2.1 P versus NP

The most prominent classes in classical complexity theory are P and NP. The class P contains all problems that can be solved in polynomial time by a deterministic Turing machine and the class NP all problems that can be solved in polynomial time by a nondeterministic Turing machine. It is widely believed that P is not equal to NP implying that there are problems in NP that are not in P. This leads to NP-hard

problems. Within the considered framework, showing that a problem is at least as hard as another problem is done by a "many-one reduction" defined as follows.

Definition 1.1. Let A and B denote two decision problems. The problem A many-one reduces to B if there is a polynomial-time computable function f such that

$$x \in A \Leftrightarrow f(x) \in B$$

for each $x \in \Sigma^*$.

A decision problem A is *NP-hard* if all problems from NP many-one reduce to A. An NP-hard problem belonging to NP is *NP-complete*. Hence, the class of NP-complete problems comprises a large set of "equivalent" problems for which presumably no polynomial-time algorithms exist.

Beyond P and NP, classical complexity theory provides classes containing problems that presumably are even harder than NP-complete problems. For example, the "oracle class" $\mathrm{P}_{||}^{\mathrm{NP}}$ consists of the problems solvable via a polynomial-time algorithm with parallel access to an NP-oracle. Interestingly, the first "natural" $\mathrm{P}_{||}^{\mathrm{NP}}$-complete problems regard the winner determination of three famous voting systems [130, 132, 183].

1.2.2 Parameterized complexity

The concept of parameterized complexity was pioneered by Downey and Fellows [76] (see also [113, 171] for text books). 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 that 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 the formal definitions.

Definition 1.2. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \Sigma^*$, where Σ is a finite alphabet. The second component is called the parameter of the problem.

We consider parameters which are positive integers or "combined" parameters which are tuples of positive integers.

Definition 1.3. A parameterized problem L is *fixed-parameter tractable* if there is an algorithm that decides in $f(p) \cdot |x|^{O(1)}$ time whether $(x, p) \in L$, where f is an arbitrary computable function depending only on p. The complexity class of all fixed-parameter tractable problems is called FPT.

We stress that the concept of fixed-parameter tractability is different from the notion of "polynomial-time solvability for constant p" since an algorithm running in $O(|x|^p)$ time does not show fixed-parameter tractability. All problem that can be solved in the running time $O(|x|^{f(p)})$ for a computable function f form the complexity class XP.

Unfortunately, not all parameterized problems are fixed-parameter tractable. To this end, Downey and Fellows [76] developed a theory of parameterized intractability by means of a completeness program with complexity classes. More specifically, the so-called *W-hierarchy* is defined by using Boolean circuits and consists of the following classes and interrelations:

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

In this work, we only provide 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 [137]. Hence, it is commonly believed that W[1]-hard problems are not fixed-parameter tractable. To show W[t]-hardness for any positive integer t, the following reduction concept was introduced.

Definition 1.4. Let $L, L' \subseteq \Sigma^* \times \Sigma^*$ be two parameterized problems. We say that L reduces to L' by a *parameterized reduction* if there are two computable functions h_1 and h_2 depending only on $|p|$ and a function f depending on x and p such that

- $(x, p) \in L \Leftrightarrow f(x, p) \in L'$ and f is computable in time $|x|^{O(1)} \cdot h_2(|p|)$
- $p' = h_1(p)$ for $(x', p') := f(x, p)$.

Analogously to the case of NP-hardness, for any positive integer t, it suffices to give a parameterized reduction from *one* W[t]-hard parameterized problem X to a parameterized problem Y to show the W[t]-hardness of Y. The 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*. For more details about parameterized complexity theory we refer to the textbooks [76, 113].

1.2.3 Multivariate algorithmics

Multivariate algorithmics can be considered as an extension of the "standard" parameterized complexity theory. While parameterized complexity provides a "two-dimensional" approach considering the input and one parameter, multivariate algorithmics allows for the investigation of more than two dimensions such as it is the case when considering combined parameters [104, 172]. Such an analysis seems to be clearly of interest for voting systems where the two "standard dimensions" votes and candidates can be combined with further parameters measuring properties of the input or the voting system. A study of for meaningful combined parameters is provided in Chapter 9.

In addition to distinguish between FPT and W[1]-hardness, one might consider the computational complexity for constant parameter values (as covered by the class XP). That is, on the one hand, a problem that is W[1]-hard with respect to a parameter p can still be solvable in polynomial time when p is a constant. One the other hand, the problem can be NP-hard even if p is a constant clearly implying W[t]-hardness for any t. Hence, within a multivariate complexity analysis, one might also consider a problem and two parameters p and q and ask whether there is a fixed-parameter algorithm with respect to p when q is constant. Such examples can be found in Subsection 9.1.2.

1.3 Algorithm design tools

We employ several techniques designed for developing fixed-parameter algorithms which will be briefly introduced in the following. We refer to the monograph by

Niedermeier [171] for a general introduction to fixed-parameter algorithms.

The corresponding running times of the algorithms given in this work are worst-case running times and usually are stated by using the "Big O"-notation:

Definition 1.5. For functions $f(x)$ and $g(x)$, $f(x)$ is in $O(g(x))$ if there are a constant c and an integer n_0 such that $f(x) \leq c \cdot g(x)$ for all $x \geq n_0$.

1.3.1 Kernelization

Kernelization is a core tool to develop parameterized algorithms [34, 76, 126, 171]. A kernelization algorithm consists of a set of *(data) reduction rules* working as follows. Given an instance $(x, p) \in \Sigma^* \times \Sigma^*$, they output in time polynomial in $|x| + |p|$ an instance $(x', p') \in \Sigma^* \times \Sigma^*$ such that the following two conditions hold.

1. (x, p) is a yes-instance if and only if (x', p') is a yes-instance.
2. $|x'| + |p'| \leq g(|p|)$ where g is a computable function.

If g is a polynomial function, then we say that the parameterized problem admits a *polynomial kernel*. We call a data reduction rule *sound* if the new instance after an application of this rule is a yes-instance iff the original instance is a yes-instance. An instance is *reduced* with respect to a reduction rule if applying the reduction rule to the instance does not change the instance.[2]

Nowadays, kernelization can be considered as a success story nicely combining theoretical analysis with practical relevance, see [34, 126] for surveys. A recent framework [35, 114] also allows to show nonexistence of polynomial kernels under some reasonable complexity-theoretic assumptions. This will be discussed in more detail in Subsection 9.2.2.

1.3.2 Size-bounded search trees

We also employ search trees for our fixed-parameter algorithms (see [171, Chapter 8] for more details). Search tree algorithms work in a recursive manner. If the algorithm solves a problem instance of size s and calls itself recursively for problem instances of sizes $s - d_1, \ldots, s - d_i$, then $(d_1, \ldots, d_i)$ is called the *branching vector* of this recursion. It corresponds to the recurrence $T_s = T_{s-d_1} + \cdots + T_{s-d_i}$ for the asymptotic size T_s of the overall search tree. Such recurrences can be solved by standard mathematical methods [171] and the asymptotic solution is determined by the roots of the so-called *characteristic polynomial* and is called *branching number*.

1.3.3 Integer linear programming

At several places in this work, we employ an approach based on integer linear programming and a famous result from Lenstra [154]. Formally, we use the following problem:

> k-Variable Integer Linear Programming Feasibility
> **Input:** A $q \times k$ matrix A with integer elements, an integer vector $b \in \mathbb{Z}^q$.
> **Question:** Is there a vector $x \in \mathbb{Z}^k$ such that $A \cdot x \leq b$?

[2] Readers familiar with voting systems but unaware of kernelization can find a simple example of a kernelization for Kemeny Score in Section 3.4.

Lenstra showed that this problem is fixed-parameter tractable with respect to the number of variables k, and this algorithmic result was later improved by several authors.

Theorem 1.1. *[154, 115, 140] The* p-Variable Integer Linear Programming Feasibility *problem can be solved using* $O(k^{2.5k+o(k)} \cdot L)$ *arithmetic operations and space polynomial in* L*, where* L *is the number of bits of the input.*

Due to Theorem 1.1 one can show fixed-parameter tractability by "formulating" a parameterized problem as an integer linear program (ILP) such that the number of variables only depends on a computable function of the considered parameter. This approach has been crucial to show fixed-parameter tractability in several areas ranging from string problems [122] to coloring [111] and graph layout [107] problems to control in elections [99].

Chapter 2

Problems and results in context

This work is concerned with a multivariate complexity analysis of several problems arising in voting. The considered problems only cover a small fraction of the whole set of "voting problems". In turn, voting can be considered as one important sub-area of social choice [6, 168] which also comprises other aspects like fair division making or judgement aggregation. Moreover, there are many works concerned with computational aspects of voting problems within the field of "computational social choice". Hence, this chapter aims at putting the considered problems and obtained results into a broader context. To this end, we highlight the following points.

- The first section gives a brief introduction into the development of voting theory. We state important issues arising in voting and display how the problems considered in this work correspond to these issues.
- The second section discusses computational aspects of voting relevant for this work.
- The third section is concerned with an overview of multivariate algorithmics results for voting problems. This includes a discussion of parameterized complexity results for voting problems.

Due to the huge amount of publications in the burgeoning field of computational social choice and in voting theory (see [53, 58, 96, 100, 103] for surveys and [6, 117, 168, 190] for text books), this chapter can only provide a necessarily incomplete picture.

2.1 A brief introduction to voting

Voting in political elections goes back to the ancient Greek. One example for the earliest form of democracy is ostracism introduced by Cleistheness about 500 years BC. It allowed the Athenians to vote for a politician they wished most for exile for 10 years. From such first attempts to give more rights to the citizens in order to prevent them from tyranny, voting developed into a fundamental tool of modern democracies. Over the last few decades, voting also gained importance in many other areas. For example, Ephrati and Rosenschein [83, 84] introduced voting to Artificial Intelligence

to solve planning problems in multiagent systems. Moreover, voting scenarios play roles in spam detection [77], in data base applications [92], in bioinformatics [138], and graph drawing [32].

In the following subsection, we very briefly discuss the development of voting and some important issues in voting as relevant to this work. Many of the "historic" voting systems are still of relevance today and some of them play a role within this work.

2.1.1 Voting rules and winner determination

The need for finding "good" systems to elect representatives of the state or of organizations such as the church led to development of a large variety of voting rules. For example, in the 13th century the writer and philosopher Ramon Llull proposed a system to elect church officials'. This can be considered as the first reported system making use of pairwise head-to-head contests [127, 167] (see Chapter 11 for a definition of Llull's system). Over the next centuries many of Llull's observations sank into oblivion. In times of the French revolution voting theory attracted new interests. In 1770, Jean-Charles de Borda proposed a "positional scoring rule" to elect members of the French Academy of Sciences [37]. In the Borda rule the positions of the candidates of a vote are converted into points and candidates with most points are declared as winners. This rule was opposed by the Marquis de Condorcet leading to a vigorous argument. The Condorcet principle from 1785 requires that a winner of an election is the candidate who is preferred to each other candidate in more than half of the votes [69]. Unfortunately, a Condorcet winner does not always exist (*Condorcet paradox*). However, Condorcet proposed a method that always elects a Condorcet winner if one exists [69]. The systems suggested by Borda and Condorcet come with different advantages and disadvantages and hence their argument remains undecided. Since we investigate scoring rules comprising the Borda systems as well as several Condorcet methods, we give two intuitive examples opposing Borda and Condorcet methods.

Many sports have world cup rankings in which a winner is "elected" based on the outcomes of a series of competitions. Examples comprise Formula 1, biathlon, and golfing. One might consider it as fair that an athlete who is better than every other athlete in more than half of the competitions should be the world champion. This is naturally reflected by the Condorcet property when considering the outcomes of the single competitions as votes over the set of athletes.

Although the Condorcet property seems quite desirable in many cases, the following example shows that this might not be true for every application. Consider a class of ten students that have to decide about a common meal. Let them have the following preferences:

- six students with: hamburger > pizza > pasta > burrito > meat balls
- four students with: pizza > pasta > burrito > meat balls > hamburger

These preferences might be interpreted as that there are six student who love hamburgers but four students detesting them. By choosing a Condorcet winner everybody would have to eat hamburger although this is the least liked meal of nearly half of the students. In contrast, the meal chosen according to the Borda rule would be pizza which seems to be a broadly acceptable choice.

The two examples illustrate that different situations demand for different voting rules. Nowadays, there is a large amount of literature concerned with investigating properties of different voting systems [38, 117, 175, 190]. This comprises famous characterization results such as a theorem saying that Kemeny is the only voting system which is neutral, Condorcet, and consistent [201]. Basically, *neutrality* means that every candidate is treated equally and *consistency* says that, if one partitions the multiset of votes into two parts and the same candidate is a winner in both parts, then this candidate also wins in the total election. Moreover, the Young theorem provides a full characterization of positional scoring rules [202]. An even stronger results is the famous Arrow's impossibility theorem showing that there is no voting system fulfilling three reasonable properties [5, 119].

Despite the huge amount of work investigating properties of voting systems, there seems to be still some gap between theory and practice in the sense that there is hardly any literature concerned with systematically "matching" voting rules to different real-life settings and in many real-world applications not the "best" voting system may be chosen. We use an example from above to illustrate the difficulty of providing general guidelines for the application of specific voting rules: In sports, such as Formula 1, one might consider it as fair to choose a Condorcet winner as world champion whenever one exists. However, the system in use is a positional scoring rule similar to Borda. Indeed, as we will see in Chapter 5 in the year 2008 this led to Lewis Hamilton becoming world champion although Felipe Massa was the Condorcet winner. However, the scoring based system rewards drivers that win a race which also might be considered desirable. In contrast, a driver could be a Condorcet winner without winning a single race.

Recall that although a Condorcet winner seems often a good choice, a practical problem is that it does not always exists. Hence, several methods have been proposed to deal with the Condorcet paradox by choosing a candidate as winner that is "closest" to a Condorcet winner. Two "edit distances" measuring the closeness have been suggested by Dodgson [71] in 1876 and Young [203] (see Chapter 6 for definitions). Another important voting rule extending the idea of Condorcet is the Kemeny rule (see Chapter 3). Herein, one looks for a "consensus preference list" minimizing the sum of distances to the input preference lists of the voters according to a natural distance measure. A winner is a top candidate of a consensus preference list.

Note that there are also many other voting rules including rules not based on preference lists such as approval voting where every voter just can vote with "yes" or "no" for every candidate [38] or rules defined over complex/overlayed preference domains such as Fallback, SP-AV [39], and TEQ [43]. All rules investigated in this work come with easy-to-understand definitions and are well-established in voting theory. In addition, they demonstrate relevance for real-world applications; for example, Kemeny's rule is used in many different fields [77, 138, 187]. Altogether, we study computational problems for positional scoring rules comprising Borda (Part II) as well as Condorcet rules, that is, voting rules electing a Condorcet winner if one exists: Dodgson, Kemeny, and Young in Part I and Copeland methods including Llull's system in Part III.

2.1.2 Strategic voting and possible winners

The previous subsection discusses the choice of a voting system in order to "maximize social welfare". Another major issue in voting theory [6] regards *strategic behavior* of the electorate, that is, can a single voter or a coalition of voters cast their votes in an insincere way such that they are better off than providing their "true" preferences. Such a behavior, denoted as *manipulation*, is considered undesirable. We refer to Faliszewski and Procaccia [103] for a recent survey on manipulation including an example showing how strategic voting results in the least favorable option to win. A voting rule is *strategy-proof* if manipulation is never beneficial for an agent. However, the famous result of Gibbard and Satterwaith shows that a reasonable strategy-proof voting rule does not exist [121, 184]. Reasonable here comes with the natural restrictions of not being a dictatorship and every candidate having a chance of winning, that is, it is a winner for at least one outcome.

On the positive side, the question whether a specific election can be manipulated in favor of a distinguished candidate can also be interpreted as follows. At a certain point of the decision process, somebody without intention to change the outcome might be interested in the question which of the candidates can still win. Going back to the Formula 1 example this reflects that a supporter of a specific driver wants to know if his favorite can still become a champion after a couple of races. This question can be further generalized to asking for a "possible winner" in the following model. Instead of having a collection of votes partitioned into a set of fully settled votes (linear orders) and the coalition coming with votes that can be cast arbitrarily, one allows for a multiset of partial preference orders. The corresponding Possible Winner problem has been introduced by Konczak and Lang [148] and is investigated in the second part of this work.

Having knowledge about the potential outcome of an election can also help to save costs by ending an election process with already determined outcome. The related task of *Preference Elicitation* (see e.g. [61, 63, 120]) can be considered as to "collect" as little information as necessary from the voters for a uniquely determined outcome. The goal is to avoid that each voter has to report his whole preference list, but to ask only for some part of the information that suffices to determine a winner.

2.1.3 Influences by external agents

Unfortunately, voting also might inspire some persons to influence the outcome of an election to their own benefits. There are numerous ways how an external agent can influence the outcome of an election in favor or disfavor of a distinguished candidate. We briefly described the settings investigated in computational social choice. We remark that the standard setting is that the external agent has complete knowledge of the preferences of all voters.

Bribery. In *bribery* the external agent can pay voters to change their preference lists in his favor [94]. There are different settings of bribery: Besides varying prices for different voters or one might allow for votes that cannot be represented as linear preference orders [93, 99]. In addition, there are more fine-grained models such as paying for specific operations, for example, single swaps of neighboring candidates in a vote, according to different price models. This leads to microbribery settings [99]

and *swap bribery* [80]. Restricting the allowed operations such that each swap must affect the distinguished candidate leads to *shift bribery* [80]. Another problem also fitting in the formal model of bribery is *campaign management* "mounting an election campaign targeted at a particular group of voters with identical preferences" [79].

Control and cloning. Bartholdi et al. [13] introduced 10 types of *control* that an external agent might apply to make his favorite candidate a winner (see also [99, 131]). The types of control comprise adding or deleting candidates and votes as well as partitioning the sets of voters or candidates. For example, a candidate can be made a winner by adding a new candidate if the new candidates take away votes from the rival. Recently, Faliszewski et al. [95] extended the scenario to the realistic setting of multimode control attacks, that is, allowing the external agents to use several types of control simultaneously. A closely related problem introduced by Elkind et al. [81] regards *cloning* where one only allows for adding candidates that are "equivalent" to one of the existing candidates. Moreover, Chevaleyre et al. [54] investigate the question whether a candidate can become a winner by adding "arbitrary" candidates.

In the third part of this work, we investigate two of the basic types of control, that is control by adding or deleting candidates. Note that while in the bribery setting usually a cost function plays a prominent role, this is not the case for control by adding or deleting candidates.

Lobbying. *Lobbying* [56, 86] arises in multi-issue referenda. Here, every voter can vote with a "yes" or "no" for a list of different topics. The external agent favors a specific outcome, that is, a list of "yes" or "no" answers. In order to achieve his goal, the agent then can pay some of the voters to cast their votes.

2.2 Computational aspects of voting

Although voting theory is a research field with a longstanding tradition and even ancient roots, systematic considerations of computational aspects started quite lately in about 1990 by a series of papers from Bartholdi, Orlin, Tovey, and Trick [10, 11, 12, 13]. The next contribution to the field of computational complexity of voting problems has been made in 1997 by Hemaspaandra et al. [130] studying the winner determination for Dodgson's rule. Starting from about 2002 there has been a rapid increase on the number of publications in this field and in computational social choice in general [53, 100].[1] In the following subsections, we describe three basic lines of research inspired by the seminal works of Bartholdi, Orlin, Tovey, and Trick also leading to the computational problems in this work. Other interesting computational aspects of voting not considered in this work comprise for example communication complexity [63] or issues in combinatorial voting [152, 198]. We refer to Chevaleyre et al. [53] for an overview of the area of computational social choice.

[1] See also the the biannual workshop series *Computational Social Choice (COMSOC)* which took place for the third time in 2010. Useful information on can also found under http://www.illc.uva.nl/COMSOC/what-is-comsoc.html.

2.2.1 Winner determination

One immediate computational issue in voting regards the computation of a winner. A voting system coming with nice properties but for which one cannot compute a winner in reasonable time is not useful in practice. While for some voting systems such as positional scoring rules the computation of a winner can be easily achieved in polynomial time, for other voting systems the computation of a winner becomes NP-hard. The most famous voting rules with NP-hard winner determination as listed in the survey [53] comprise Banks, Dodgson, Kemeny, Slater, and Young. We propose a parameterized complexity analysis as a possible way out of the dilemma and provide such studies for Dodgson, Kemeny, and Young in the first part of this work. Remarkably, the winner determination under these three rules led to the first natural problems that are $\mathrm{P}_{||}^{\mathrm{NP}}$-complete [130, 132, 183].

2.2.2 Manipulation and possible winners

As discussed in Subsection 2.1.2, a key issue in voting theory regards the undesirable property of manipulation. Since according to the Gibbard-Satterwaith theorem [121, 184] there is no reasonable strategy-proof voting rule, Bartholdi et al. [11] suggested computational hardness as a way out. The simple but brilliant idea is that when a voting rule can be manipulated in general but it is computationally difficult to find out how to cast the votes to achieve the desired goal, a voting rule cannot be manipulated in practice. This leads to the following basic computational problem for any voting rule r.

> Manipulation
> *Input:* An election (V, C), a coalition size k, a distinguished candidate $c \in C$.
> *Question:* Is there a size-k multiset V' of votes on C such that c is a winner according to r in $(V \cup V', C)$?

There are many variants of Manipulation such as having weighted voters or destructive manipulation where the goal is to prevent a candidate from winning (see e.g. [65, 103]). Bartholdi, Orlin, Tovey, and Trick focused on the special case of having a coalition of size one. After obtaining polynomial-time solvability results for manipulating a set of common voting rules [11], Bartholdi and Orlin [10] identified a known voting rule for which Manipulation becomes NP-hard even for a single manipulator. Nowadays, there is a huge amount of literature on the manipulation problem including discussions about worst-case versus average-case hardness or frequency of success/hardness (e.g. [87, 116, 129, 179, 195, 196]). For many common voting rules the classical computational complexity has been explored (e.g. [46, 65, 129, 200, 204]). However, there are still interesting open questions. In particular, whether Manipulation is NP-hard for Borda is a prominent open question [103, 199, 200]. We also refer to Faliszewski and Procaccia [103] for a recent survey on Manipulation.

Konczak and Lang [148] introduced the Possible Winner problem directly generalizing Manipulation (see Subsection 2.1.2) and obtained first computational complexity results for several common voting rules. Their results have been extended by a series of publications (see Chapter 7 for more details). The second part of this work is concerned with a multivariate complexity analysis of Possible Winner for positional scoring rules.

2.2.3 Control

Similar to the idea of preventing strategic voting, Bartholdi et al. [13] suggested computational hardness as a "shield" against external control of an election (see Subsection 2.1.3). Today, many works provide hardness results based on worst-case complexity for many different types of voting rules [88, 90, 99, 131, 157, 169] (see also Chapter 11) including extended settings such as multi-mode attacks [95]. However, the corresponding NP-hardness results do not hold for all realistic settings. For example, considering so-called single-peaked preferences, one can decide for many voting rules in polynomial time whether it is possible to control (or manipulate) an election [41, 101]. In the third part of this thesis, we consider control by adding or deleting candidates from a parameterized complexity point of view, showing that (from a worst-case perspective) the computational hardness still holds for realistic settings such as adding or deleting only a bounded number of candidates.

2.3 Multivariate algorithmic results

Voting problems naturally come with meaningful parameters such as the numbers of candidates or votes, the size of the coalition in Manipulation, or the number of added candidates when controlling an election. Although previous studies already provided some results in this direction, until recently, there were no systematic investigations with respect to the multivariate computational complexity of voting problems. In particular, the first work explicitly concerned with parameterized complexity was provided in 2007 by Christian et al. [56] and a survey by Lindner and Rothe [156] from 2008 comprises the papers [20, 26, 29, 56, 99] concerned with parameterized algorithmics for voting problems (three of them being part of this thesis). In the following, we discuss some famous examples for multivariate algorithmic results as well as all parameterized complexity results for voting problems of which we are aware. We explicitly include the results from this thesis and the corresponding papers since they are "interlaced" with many of the other works. Omitting them would yield an incomplete picture for some of the problems. The results are ordered according to the type of problem.

Results for winner determination. Besides showing the NP-hardness of determining a Dodgson winner (see Chapter 6 for a definition), Bartholdi et al. [12] provided an integer linear program for which the number of variables is bounded by the number of candidates. They also deduced polynomial-time solvability for a constant number of candidates by using a result from Lenstra [154]. As discussed in Subsection 1.3.3, this also implies fixed-parameter tractability. In this sense, Bartholdi et al. provided a fixed-parameter tractability result before this concept had been introduced with Downey and Fellows groundbreaking monograph [76]. Some improvements of the integer linear program also leading to fixed-parameter tractability have been devised by McCabe-Dansted [163]. Similarly, fixed-parameter tractability with respect to the number of candidates for determining a Young winner follows from an ILP provided by Young [203] in combination with Lenstra's result. We devised the parameterized complexity for Dodgson and Young elections with respect to problem-specific parameterizations measuring corresponding "edit distances" from being a Condorcet winner [26, 27] (see Chapter 6). Our results comprise a dynamic programming algo-

rithm as well as parameterized reductions. Recent work [106] further extended our results for Dodgson by showing W[1]-hardness with respect to the number of votes and providing evidence for the nonexistence of a polynomial kernel with respect to the edit distance.

Elkind et al. [82] devised the following "general" result. Using Lenstra's theorem, they showed fixed-parameter tractability with respect to the number of candidates for the winner determination for a broad class of voting rules defined under a so-called "distance rationalizability framework". This can be considered as a generalization of some of the previous results. For example, Dodgson's rule is contained in the considered class of voting rules.

For computing Kemeny rankings, we initiated a parameterized complexity analysis with respect to several meaningful parameterizations [20, 22, 21] (see Chapter 3). Meanwhile, these results have been extended and partially improved in several works [142, 110, 189] (see Section 3.8 for a detailed discussion). In further work [24] we extended the range of techniques by providing polynomial-time data reduction rules, also leading to fixed-parameter tractability. Finally, we experimentally showed their practical usefulness [16].

Some voting systems are naturally reflected by problems on (directed) graphs. For example, the Slater Ranking problem (see e.g. [57]) is the same problem as Feedback Arc Set for which a range of parameterized algorithmic results exist [3, 52, 72]. A further example regards tournament solutions that can be understand as selecting a set of winners based on a complete and asymmetric relation on a set of candidates. Some tournament solutions, such as Banks or the Tournament Equilibrium Set, are of particular interest in computational social choice [42]. Very recently, Brandt et al. [42] identified a natural parameterization for tournament solutions that are "composition-consistent" and provided corresponding fixed-parameter tractability results.

Results for possible winner determination. Part II of this work comprises a multivariate complexity analysis of Possible Winner for positional scoring rules based on the publications [15, 18, 19, 28]. Previous to these publications, Xia and Conitzer [194] provided an NP-hardness results holding even when a parameter measuring the amount of incompleteness is of constant value (see Chapter 8 for more details).

Results for manipulation. Conitzer et al. [65] investigated the computational complexity of Manipulation depending on the number of candidates for nine common voting rules. For most of the considered rules they provided a small positive integer s such that weighted Manipulation is NP-hard if and only if the number of candidates is at least s. Hemaspaandra and Hemaspaandra [129] obtained a full characterization of the computational complexity for weighted Manipulation for all scoring rules for a fixed number of candidates; they mainly obtained NP-hardness. On the positive side, Conitzer et al. [65] show fixed-parameter tractability with respect to the number m of candidates for manipulating the "Single Transferable Vote (STV)" rule by one voter. The corresponding algorithm runs in $1.62^m \cdot \text{poly}$ time.

Manipulation for STV is NP-hard even for coalition size one [10]. Hence, there is no hope for fixed-parameter tractability with respect to the parameter coalition size. The same holds true for several other common voting systems for which Manipula-

TION becomes NP-hard for every constant coalition size greater than one [102, 200].

Results for bribery. In the scenario of SWAP BRIBERY, each voter assigns a certain price for swapping the position of two consecutive candidates in his preference list. Very recently, Dorn and Schlotter [74] investigated the parameterized complexity of SWAP BRIBERY with respect to several natural parameters. They provide classical and parameterized hardness results as well as fixed-parameter tractability results. In particular, they give problem kernelizations for two combined parameters.

Results for control. Faliszewski et al. [99] devised a systematic analysis of the computational complexity of controlling Copeland voting systems. One of their main contributions is to show NP-hardness for all previously studied types of control for the constructive case. In addition, they obtained fixed-parameter tractability results with respect to the number of votes or with respect to the number of candidates for some types of control. In Chapter 11, we extend the study of control in Copeland and plurality voting for control by adding and deleting candidates. More specifically, we investigate the parameterized complexity with respect to the number of added/deleted candidates. Some additional results in this direction (partially "reproving" already known results) have been provided by Liu et al. [157]. Recently, Erdély and Fellows [89] considered the parameterized complexity of control by adding/deleting candidates or votes for fallback voting, a voting system introduced by Brams and Sanver [39]. An other recent work [158] investigates the parameterized complexity of control for maximin.

Results for lobbying. The first work (explicitly) concerned with parameterized complexity for voting problems was provided by Christian et al. [56], showing W[2]-completeness for LOBBYING. Erdély et al. [85] further extended the lobbying problem to a probabilistic setting and besides other results provided several fixed-parameter tractability as well as W[2]-completeness results.

Part I

Winner Determination

We accomplish a multivariate complexity analysis for NP-hard problems corresponding to the winner determinination for three famous voting systems going back to Kemeny, Dodgson, and Young, respectively. Herein, the Kemeny voting system plays the most important role and the corresponding results comprise three chapters. Kemeny's voting system is concerned with optimally aggregating a multiset of preference lists into a consensus list. In Chapter 3, we study the parameterized complexity of the underlying decision problem Kemeny Score with respect to several different parameterizations. This includes a discussion about the relevance and motivation of the parameterizations illustrating the usefulness of a multivariate complexity analysis for voting problems. Chapter 4 further extends this study by introducing the concept of "partial kernelization". Based on this, we devise an additional result for the parameter measuring the average distance between pairs of input votes. Moreover, the partial kernelization concept naturally leads to additional parameterizations. In Chapter 5, we provide experimental results for computing optimal Kemeny rankings.
The last chapter of this part focuses on Dodgson and Young elections. In both voting systems the score of a candidate measures the "distance" from a Condorcet winner using different editing operations. We oppose the parameterized complexities of these two voting systems by showing fixed-parameter tractability with respect to the score for Dogson and W[2]-completeness with respect to the score for Young.

Chapter 3

Parameterizations for Kemeny

The problem of aggregating several preference lists into a consensus list is naturally modelled by an election system that goes back to Kemeny: Given a multiset of preference lists, one searches for a preference list such that the sum of the "distances" from this preference list to all remaining preference lists is minimized. Herein, the distance between two votes is measured by the number of inversions. This chapter is based on the paper [21] which initiated a multivariate complexity analysis for computing optimal Kemeny rankings by systematically identifying and analyzing meaningful parameterizations.

Besides the three obvious parameters "number of votes", "number of candidates", and "solution size" (called Kemeny score), we consider further structural parameterizations. More specifically, we show that the Kemeny score (and a corresponding Kemeny consensus) of an election can be computed efficiently whenever the *average* pairwise distance between two input votes is not too large. In other words, Kemeny Score is fixed-parameter tractable with respect to the parameter "average pairwise Kendall-Tau distance" d_a. Moreover, we extend our studies to the parameters "maximum range" and "average range" of positions a candidate takes in the input votes. Whereas Kemeny Score remains fixed-parameter tractable with respect to the parameter "maximum range", it becomes NP-complete in case of an average range of two. Finally, we extend some of our results to votes with ties and incomplete votes, where in both cases one no longer has permutations as input. As discussed in the corresponding sections of this chapter, several of our results from [21] meanwhile have been improved [142, 189] and the range of parameterizations has been extended [161].

3.1 Introduction

Kemeny's voting scheme goes back to the year 1959 [143] and was later specified by Levenglick [155]. It can be described as follows. A "Kemeny consensus" is a preference list that is "closest" to the preference lists provided by the votes: For each pair of votes v, w, the so-called *Kendall-Tau distance* (*KT-distance* for short) between v and w, also known as the number of inversions between two permutations,

is defined as

$$\mathrm{dist}(v, w) = \sum_{\{a,b\} \subseteq C} d_{v,w}(a, b),$$

where the sum is taken over all unordered pairs $\{a, b\}$ of candidates, and $d_{v,w}(a, b)$ is set to 0 if v and w rank a and b in the same order, and is set to 1, otherwise. Using divide and conquer, the KT-distance can be computed in $O(m \cdot \log m)$ time (see, e.g., [145]). The *score* of a preference list l with respect to an election (V, C) is defined as

$$\sum_{v \in V} \mathrm{dist}(l, v).$$

A preference list l with the minimum score is called *Kemeny consensus* (or Kemeny ranking) of (V, C) and its score $\sum_{v \in V} \mathrm{dist}(l, v)$ is the *Kemeny score* of (V, C). The central problem considered in this and the following two chapters is as follows:

Kemeny Score
Input: An election (V, C) and a positive integer k.
Question: Is the Kemeny score of (V, C) at most k?

All our algorithms do not only decide Kemeny Score but can also compute a corresponding Kemeny consensus. The computation of a Kemeny consensus has numerous applications, ranging from building meta-search engines for the web or spam detection [77] over databases [92, 187] to the construction of genetic maps in bioinformatics [138]. Kemeny rankings are also desirable in classical voting scenarios such as the determination of a president (see, for example, `www.votefair.org`) or the selection of the best qualified candidates for job openings. The wide range of applications is due to the fulfillment of many desirable properties from the social choice point of view. For example, Kemeny's voting system is the only voting system which is neutral, consistent, and Condorcet [201]. A Kemeny consensus can also be considered as "maximum likelihood estimator" under the assumption that there is an unobserved "correct" ranking and the votes are noisy estimates of this [60] (see also [69, 62] for work in this direction).

We briefly mention that Kemeny Score is closely related to Weighted Feedback Arc Set, where, given a directed graph with edge weights, one seeks for a minimum-weight set of arcs whose deletion leads to an acyclic graph. The unweighted version Feedback Arc Set directly corresponds to a voting system proposed by Slater for which the computational complexity has been investigated by Conitzer [57].

Related work. Bartholdi et al. [12] showed that Kemeny Score is NP-complete. This remains true even when restricted to instances with only four votes [77, 78]. The corresponding proof contained a small error which was fixed by Biedl et al. [32]. Hemaspaandra et al. [132] provided further, exact classifications of the classical computational complexity of Kemeny elections. More specifically, while Kemeny Score is NP-complete, they provided $\mathbf{P}_{\|}^{\mathbf{NP}}$-completeness results for other, more general versions of the problem.

Given the computational hardness of Kemeny Score on the one hand and its practical relevance on the other hand, polynomial-time approximation algorithms have been studied. The Kemeny score can be approximated to a factor of 8/5 by a deterministic algorithm [192] and to a factor of 11/7 by a randomized algorithm [2].

Table 3.1: Overview of the main results provided in this chapter. To make the running times easier to read, we use the O^*-notation that allows to omit polynomial factors.

Parameter	Running time	
Number of candidates m	$O^*(2^m)$	(Section 3.3)
Kemeny score k	$O^*(1.53^k)$	(Section 3.4)
Maximum range of candidate positions r	$O^*(32^r)$	(Subsection 3.6.1)
Average range of candidate positions r_a	NP-c for $r_a \geq 2$	(Subsection 3.6.2)
Average KT-distance d_a	$O^*(16^{d_a})$	(Section 3.5)

A polynomial-time approximation scheme (PTAS) has been developed by Kenyon-Mathieu and Schudy [144]. However, its running time is impractical. Schalekamp and van Zuylen [185] experimentally evaluated the quality of different approximation algorithms and heuristics.

Regarding the computation of an *exact* Kemeny consensus, Conitzer, Davenport, and Kalagnanam [68, 59] performed computational studies using heuristic approaches such as greedy and branch-and-bound. These experimental investigations focus on computing strong admissible bounds for speeding up search-based heuristic algorithms. In contrast, our focus is on exact algorithms with provable asymptotic running time bounds for the developed algorithms.

Regarding parameterized complexity of Kemeny Score, we initiated a study in the conference article [20] which was further extended in the conference article [22]. These two articles were combined to the journal article [21]. Recent work [142, 161, 189] improved some of our results and investigated additional parameterizations. Specific results will be discussed and opposed to our results in the corresponding sections. Section 3.8 provides an overview of the state of the art. The following subsection introduces the parameterization and results from [21] described in this chapter. In addition, the following two chapters extend this study by developing data reduction rules (Chapter 4) and providing an experimental analysis of some of our fixed-parameter algorithms (Chapter 5).

3.2 Parameterizations and our results.

This section overviews the considered parameterizations and our corresponding results. We also discuss the "art" of finding different, practically relevant parameterizations of Kemeny Score. Besides the considerations for the three obvious parameters "number of votes", "number of candidates", and "Kemeny score", our paper focusses on structural parameterizations, that is, structural properties of input instances that may be exploited to develop efficient algorithms. Our results are summarized in Table 3.1 and will be explained in the following. In addition, we extend some of our findings to the cases where ties within votes are allowed and to incomplete votes where not all candidates are ranked (see Section 3.7). Note that studying the parameter "number of

votes" is pointless because the problem is already NP-complete for only four votes [77].

Number of candidates. By trying all possible permutations of the m candidates, one trivially attains an efficient algorithm if m is very small. The corresponding combinatorial explosion $m!$ in the parameter is fairly large, though. Using dynamic programming, we can improve this to an algorithm running in $O(2^m \cdot m^2 \cdot n)$ time where n denotes the number of votes.

Kemeny Score. For the Kemeny score k, we derive an algorithm solving Kemeny Score in $O(1.53^k + m^2 n)$ time. This algorithm is based on a problem kernelization and a depth-bounded search tree. This may be considered as the "canonical parameterization" because the parameter measures the "solution size"—however, it is conceivable that in many applications the value of k may not be small.

Maximum and average KT-distance. The maximum KT-distance of an election (V, C) is defined as

$$d_{\max} := \max_{u,v \in V, u \neq v} \operatorname{dist}(u, v).$$

Analogously, the average KT-distance d_a is defined as

$$d_a := \frac{1}{n(n-1)} \cdot \sum_{u,v \in V, u \neq v} \operatorname{dist}(u, v).$$

Since $d_{\max} \geq d_a$, any fixed-parameter algorithm with respect to d_a is also a fixed-parameter algorithm with respect to $d_{\max}$. Fixed-parameter tractability with respect to the maximum KT-distance allows for an efficient computation of a Kemeny consensus whenever every pair of votes is similar. For the parameter average KT-distance, an efficient computation is possible whenever the votes are similar enough *on average*. Thus, it can also cope with "outlier votes".

Let us briefly discuss the naturalness of average parameterization for the computation of Kemeny rankings. It is plausible that the aggregation of several preference lists is only meaningful when the given input lists have a sufficiently high degree of average similarity. Otherwise the median consensus found may be meaningless since it tries to fit strongly opposing demands. Studying the parameterization with respect to the average KT-distance also naturally reflects a view on voting proposed by Conitzer and Sandholm [62]. More specifically, they pointed out that one potential view of voting is that there exists a "correct" outcome (ranking), and each vote corresponds to a noisy perception of this correct outcome. Some experimental studies [57, 59] observed that for computing a Kemeny consensus the running time increases if the input instances become noisier. Showing fixed-parameter tractability with respect to the average KT-distance hence can also be considered as an explanation for this experimentally observed behavior.

We show fixed-parameter tractability with respect to the "average KT-distance" by providing a dynamic programming algorithm running with an exponential worst-case running time factor of 16^{d_a}. Note that in the meantime, this factor has been improved to 5.83^{d_a} by Simjour [189] and asymptotically further improved to $2^{O(\sqrt{d_a})}$ by Karpinski and Schudy [142]. Moreover, in Chapter 4, we provide polynomial-time data reduction rules also leading to fixed-parameter tractability with respect to d_a.

Maximum and average candidate range. We introduce structural parameterizations based on the range of positions a candidate can assume. Let the *position* of a candidate c in a vote v be the number of candidates that are better than c in v. That is, the leftmost (and best) candidate in v has position 0 and the rightmost has position $m-1$. Let $v(c)$ denote the position of c in v. For an election (V, C) and $c \in C$, the *range* of c is defined as

$$r(c) := \max_{v,w \in V} \{|v(c) - w(c)|\} + 1.$$

The maximum range $r_{\max}$ of an election is defined as $r_{\max} := \max_{c \in C} r(c)$ and the average range r_a is defined as

$$r_a := \frac{1}{m} \sum_{c \in C} r(c).$$

We show that Kemeny Score can be solved in $O(32^{r_{\max}} \cdot (r_{\max}^2 \cdot m + r_{\max} \cdot m^2 \log m \cdot n) + n^2 \cdot m \log m)$ time by a dynamic programming approach. If we study the parameter *average* range instead of *maximum* range, Kemeny Score turns NP-complete already for constant parameter values.

The parameterization by position range might reflect the situation that whereas voters can be more or less decided concerning groups of candidates (e.g., political parties), they may be quite undecided and, thus, unpredictable, concerning the ranking within these groups. If these groups are small this can also imply small range values, thus making the quest for a fixed-parameter algorithm in terms of range parameterization attractive.

3.2.1 Range versus distance parameterizations

The following two concrete scenarios underpin the usefulness of "range" *and* "KT-distance based" parameterizations. First, consider a survey about the prestige of different car brands. A likely assumption is that every brand will only appear at a certain "range" in all preference lists since all participants (voters) have a similar opinion about every car brand (candidate). Hence, this may lead to an election with a small maximum range. Second, let us change the question of the survey from the "most prestigious car brand" into the "kind of car the participants would like to drive". For this question, it might happen that the same kind of car is in the top position in one preference list (maybe because it is the fastest one) and in the last position in another preference list (maybe because it also has the highest mileage). Then, we cannot assume to have a small maximum range. However, it is still reasonable that on average the participants have similar preferences, for example, most participants would prefer to drive a "Mercedes" to a "Škoda" because they care more about prestige than about pollution.

In what follows, we provide evidence that the value of the "average KT-distance" can significantly differ from the "maximum/average range of position" for an election. More specifically, there are input instances of Kemeny Score having a small range value and a large average KT-distance, and *vice versa*. This justifies separate investigations for both parameterizations.

v_1	:	$a > b > c > d > e > f > \dots$
		$\vdots$
v_i	:	$a > b > c > d > e > f > \dots$
v_{i+1}	:	$b > a > d > c > f > e > \dots$
		$\vdots$
v_{2i}	:	$b > a > d > c > f > e > \dots$

Figure 3.1: Election with small maximum range but large average KT-distance.

v_1	:	a	$>$	b	$>$	c	$>$	d	$>$	e	$>$	f	$>$	$\dots$
v_2	:	b	$>$	c	$>$	d	$>$	e	$>$	f	$>$	$\dots$	$>$	a
v_1'	:	a	$>$	b	$>$	c	$>$	d	$>$	e	$>$	f	$>$	$\dots$
		$\vdots$												

Figure 3.2: Election with small average KT-distance but large maximum range.

First, we provide an example where one can observe a small maximum candidate range whereas one has a large average KT-distance, see Figure 3.1. The election in Figure 3.1 consists of $n = 2i$ votes such that there are two groups of i identical votes. The votes of the second group are obtained from the first group by swapping neighboring pairs of candidates. Clearly, the maximum range of candidates is two. However, for m candidates the average KT-distance d_a is

$$d_a = \frac{2 \cdot (n/2)^2 \cdot (m/2)}{n(n-1)} > m/4$$

and, thus, d_a is unbounded for an unbounded number of candidates.

Observation 3.1. *There are elections with maximum range two and an unbounded average KT-distance.*

Second, we present a simple example where the average KT-distance is small but the maximum range of candidates is large, see Figure 3.2. In the election of Figure 3.2 all votes are equal except that candidate a is at the last position in the second vote, but on the first position in all other votes. The maximum range equals the range of candidate a and hence the number of candidates. In contrast, by adding a sufficient number of copies of the first vote the average KT-distance can be made smaller than one.

Observation 3.2. *There are elections with average KT-distance smaller than one and an unbounded maximum range.*

With some effort we can extend this observation from maximum to average range. To this end, construct an election consisting of m candidates and the following votes. Let V_m be a set of m votes such that every candidate is in one of the votes at the first and in one of the votes at the last position; the remaining positions can be filled

arbitrarily. Then, for some $N > m^3$, add N further votes V_N in which all candidates have the same arbitrary order. Let $D(V_m)$ ($D(V_N)$) be the average KT-distance within the votes of V_m (V_N) and $D(V_N, V_m)$ be the average KT-distance between pairs of votes with one vote from V_N and the other vote from V_m. Since m^2 is an upper bound for the pairwise (and average) KT-distance between any two votes, it holds that $D(V_m) \leq m^2$, $D(V_N) = 1$, and $D(V_N, V_m) \leq m^2$. Moreover, we have $m \cdot (m-1)$ ordered pairs of votes within V_m, $N \cdot m$ pairs between V_N and V_m, and $N \cdot (N-1)$ pairs within V_N. Since $N > m^3$ it follows that

$$d_a \leq \frac{m(m-1) \cdot m^2 + Nm \cdot m^2 + N(N-1) \cdot 1}{N(N-1)} \leq 3.$$

In contrast, the range of every candidate is m, thus the average range is m.

Observation 3.3. *There are elections with average KT-distance at most three but an unbounded average range.*

3.3 Parameterization by the number of candidates

Simply trying all $m!$ permutations of candidates already leads to the fixed-parameter tractability of Kemeny Score with respect to the "number of candidates" m. By means of dynamic programming, we improve this to an algorithm running in $O(2^m \cdot m^2 \cdot n)$ time. The same result can also be obtained by a reduction to Feedback Arc Set on Tournaments [78] and a corresponding exact algorithm [181]. In the following, we briefly sketch the algorithm that is implemented and experimentally evaluated in Chapter 5.

Dynamic programming algorithm. For each subset $C' \subseteq C$ compute the Kemeny score of the given election (V, C) restricted to C'. The recurrence for a given subset C' is based on considering every subset $C'' \subseteq C'$ where C'' is obtained by deleting a single candidate c from C'. Let l'' be a Kemeny consensus for the election system restricted to C''. For every c from C', compute the score of the permutation l' of C' obtained from l'' by putting c in the first position and take the minimum score over all corresponding possibilities as score for C'. Then, it is not hard to observe that this score is the Kemeny score of the election restricted to C'. We refer to [22] for a formal proof leading to the following.

Theorem 3.1. Kemeny Score *can be solved in* $O(2^m \cdot m^2 \cdot n)$ *time.*

3.4 Parameterization by the Kemeny score

In this section, we show that Kemeny Score is fixed-parameter tractable with respect to the Kemeny score k. Our result is based on a kernelization and a search tree algorithm. This result has been further improved in recent work [189, 142] by using different approaches (see Section 3.8 for more details). In the following, we sketch our results anyway since they are useful to understand some basic properties of the Kemeny score problem and the performance of the search tree algorithm will be experimentally evaluated in Chapter 5.

Our results rely on the following lemma, whose correctness directly follows from the extended Condorcet criterion [191].

Lemma 3.1. *Let a and b be two candidates. If $a > b$ in all votes, then every Kemeny consensus fulfils "$a > b$".*

Candidate pairs for which the relative order in a Kemeny consensus is not determined by Lemma 3.1 are called *conflict pairs*.

Problem Kernel. When applied to an input instance (V, C, k) of Kemeny Score, the following polynomial-time executable data reduction rules yield an "equivalent" election with at most $2k$ candidates and at most $2k$ votes. We first describe a reduction rule reducing the number of candidates and then present a reduction rule reducing the number of votes.

Rule 3.1. Delete every candidates that is in no conflict pair.

The soundness of Rule 3.1 can be seen as follows. If a candidate c is not part of a conflict pair, then there exists a partition of the remaining candidates into two subsets C_1 and C_2 such that in all votes all candidates $c_1 \in C_1$ are positioned better than c and all candidates $c_2 \in C_2$ are positioned worse than c. Thus, due to Lemma 3.1, the position of c in every Kemeny consensus is already determined and the removal of c does not affect the Kemeny score.

We apply a second simple data reduction rule to get rid of too many identical votes.

Rule 3.2. If there are more than k votes in V identical to a preference list l, then return "yes" if the score of l is at most k; otherwise, return "no".

To see the soundness of Rule 3.2, assume that we have a Kemeny consensus that is not identical to l. Then, the KT-distance between l and this Kemeny consensus is at least 1. Since we have more than k copies of l, the Kemeny score exceeds k, a contradiction.

Based on the two reduction rules, we achieve the following result.

Theorem 3.2. Kemeny Score *admits a problem kernel with at most $2k$ votes over at most $2k$ candidates. It can be computed in $O(m^2 n)$ time for m candidates and n votes.*

Proof. After applying Rule 3.1, every remaining candidate is part of at least one conflict pair. If there are more than $2k$ such candidates, then they must form more than k conflict pairs. For each conflict pair, there are two possibilities to place the candidates of this pair in any preference list. For both possibilities, the score of this preference list is increased by at least one, implying that, with more than k dirty pairs, there is no preference list with a score at most k. Hence, after having exhaustively applied Rule 3.1, there are at most $2k$ candidates.

The bound on the number of votes is achieved as follows. Between two distinct preference lists, the KT-distance is at least 1. After applying Rule 3.2, a preference list has at most k "copies" in V. Therefore, if $n > 2k$, then the score of every preference list is at least $k+1$. Hence, after having exhaustively applied Rule 3.2, in a yes-instance there are at most $2k$ votes. The running time of both rules is easy to see [22]. □

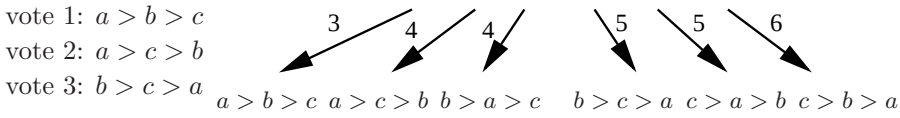


Figure 3.3: An illustration of the branching for the conflict triple $\{a, b, c\}$ in the election given by votes 1, 2, and 3. For the six orders of the three candidates a, b, and c one can reduce the Kemeny score at the search tree node by the amount depicted next to the corresponding arrow. For example, for the leftmost possibility, the score of "$a > b > c$" is three since its distance to vote 1 is zero, the distance to vote 2 is one, and the distance to vote 3 is two.

Search tree algorithm. Branching into conflict pairs directly leads to a search tree of size at most 2^k: At each search tree node we can branch into the two possible relative orders of a conflict pair and in each case we can decrease the parameter at least by one. For all remaining candidate pairs their relative order is already fixed due to Lemma 3.1. We briefly discuss how a more refined branching leads to a search tree of smaller size. Herein, the basic idea is to branch into "conflict tuples" consisting of more than two candidates.

Definition 3.1. For a candidate subset $C_s \subseteq C$ consisting of s candidates, consider the auxiliary graph having one vertex for every $c_i \in C_s$ and an edge between the vertices corresponding to $c_i, c_j \in C_s$ if c_i and c_j form a conflict pair. If the constructed graph is connected, then C_s forms a *conflict s-tuple*.

An example for branching into conflict triples is provided in Figure 3.3. The corresponding branching vector is $(6, 5, 5, 4, 4, 3)$ yielding the branching number 1.52. After branching into all conflict s-tuples for a fixed $s \geq 2$, it remains to determine the relative order within all conflict tuples of size smaller than s. Since two candidates corresponding to disjoint conflict tuples cannot form a conflict pair, the order of possibly remaining conflict tuples can be decided locally in $O(s!)$ time [22, 44]. By a systematic analysis of the case distinctions based on conflict triples and making use of the data reduction rules, one can show the following.

Theorem 3.3 ([22]). Kemeny Score *can be solved in $O(1.53^k + m^2 n)$ time.*

Bredereck [44] showed that branching into four-tuples further improves the search tree size to $O(1.508^k)$. The further improvements are based on different approaches [189, 142]. The above search tree algorithm also works for instances in which the votes are weighted by positive integers. More specifically, one can use exactly the same search tree, but may gain some further (heuristic) speed-up by decreasing the parameter value according to the weights.

3.5 Parameterization by the average KT-distance

In this section, we further extend the range of parameterizations by providing a fixed-parameter algorithm with respect to the parameter "average KT-distance" d_a. We start with showing how the average KT-distance can be used to provide an upper bound on the range of positions that a candidate can take in any optimal Kemeny

consensus. Based on this crucial observation, we then state the algorithm running in $16^{d_a} \cdot \text{poly}(n, m)$ time. The currently asymptotically fastest fixed-parameter algorithm with respect to d_a is provided by Karpinski and Schudy [142] and runs in $2^{O(\sqrt{d_a})} \cdot \text{poly}(n, m)$ time. We provide our result anyway since, first, the following Observation (Subsection 3.5.1) seems to be of independent interest and, second, the dynamic programming procedure (Subsection 3.5.2) is needed in exactly the same form to show fixed-parameter tractability with respect to the "maximum range" of a candidate.

Within this section, let $d := \lceil d_a \rceil$. For an election (V, C) and a candidate $c \in C$, the *average position* $p_a(c)$ of c is defined as

$$p_a(c) := \frac{1}{n} \cdot \sum_{v \in V} v(c).$$

3.5.1 A crucial observation

Our fixed-parameter tractability result with respect to the "average KT-distance" is based on the following lemma establishing a connection between the average position of a candidate and its position in a Kemeny consensus. Note that the recent fixed-parameter algorithms with respect to d_a [189, 142] do not provide results in this direction.

Lemma 3.2. *Let d_a be the average KT-distance of an election (V, C) and $d = \lceil d_a \rceil$. Then, in every Kemeny consensus l, for every candidate $c \in C$ with average position $p_a(c)$, it holds that $p_a(c) - d < l(c) < p_a(c) + d$.*

Proof. The proof is by contradiction and consists of two claims: First, we show that we can find a vote with Kemeny score less than $d \cdot n$, that is, the Kemeny score of the instance is less than $d \cdot n$. Second, we show that in every Kemeny consensus every candidate is in the claimed range. More specifically, we prove that every consensus in which the position of a candidate is not in a "range d of its average position" has a Kemeny score greater than $d \cdot n$, a contradiction to the first claim.

Claim 1: K-score$(V, C) < d \cdot n$.

Proof of Claim 1: To prove Claim 1, we show that there is a vote $v \in V$ with $\sum_{w \in V} \text{dist}(v, w) < d \cdot n$, implying this upper bound for an optimal Kemeny consensus as well. By definition,

$$d_a = \frac{1}{n(n-1)} \cdot \sum_{v, w \in V, v \neq w} \text{dist}(v, w) \tag{3.1}$$

$$\Rightarrow \exists v \in V \text{ with } d_a \geq \frac{1}{n(n-1)} \cdot n \cdot \sum_{w \in V, v \neq w} \text{dist}(v, w) \tag{3.2}$$

$$= \frac{1}{n-1} \cdot \sum_{w \in V, v \neq w} \text{dist}(v, w) \tag{3.3}$$

$$\Rightarrow \exists v \in V \text{ with } d_a \cdot n > \sum_{w \in V, v \neq w} \text{dist}(v, w). \tag{3.4}$$

Since we have $d = \lceil d_a \rceil$, Claim 1 follows directly from Inequality (3.4).

The next claim shows the given bound on the range of possible candidates positions.

Claim 2: In every optimal Kemeny consensus l, every candidate $c \in C$ fulfills $p_a(c) - d < l(c) < p_a(c) + d$.

Proof of Claim 2: We start by showing that, for every candidate $c \in C$, we have

$$\text{K-score}(V, C) \geq \sum_{v \in V} |l(c) - v(c)|. \tag{3.5}$$

Note that, for every candidate $c \in C$, for two votes v, w we must have $\text{dist}(v, w) \geq |v(c) - w(c)|$. Without loss of generality, assume that $v(c) > w(c)$. Then, there must be at least $v(c) - w(c)$ candidates that have a smaller position than c in v and that have a greater position than c in w. Further, each of these candidates increases the value of $\text{dist}(v, w)$ by one. Based on this, Inequality (3.5) directly follows as, by definition, $K\text{-score}(V, C) = \sum_{v \in V} \text{dist}(v, l)$.

To simplify the proof of Claim 2, in the following, we shift the positions in l such that $l(c) = 0$. Accordingly, we shift the positions in all votes in V, that is, for every $v \in V$ and every $a \in C$, we decrease $v(a)$ by the original value of $l(c)$. Clearly, shifting all positions does not affect the relative differences of positions between two candidates. Then, let the set of votes in which c has a nonnegative position be V^+ and let V^- denote the remaining set of votes, that is, $V^- := V \backslash V^+$.

Now, we show that if candidate c is placed outside of the given range in an optimal Kemeny consensus l, then $\text{K-score}(V, C) > d \cdot n$. The proof is by contradiction. We distinguish two cases:

Case 1: $l(c) \geq p_a(c) + d$.
As $l(c) = 0$, in this case $p_a(c)$ becomes negative. Then,

$$0 \geq p_a(c) + d \Leftrightarrow -p_a(c) \geq d.$$

It follows that $|p_a(c)| \geq d$. The following shows that Claim 2 holds for this case.

$$\sum_{v \in V} |l(c) - v(c)| = \sum_{v \in V} |v(c)| \tag{3.6}$$

$$= \sum_{v \in V^+} |v(c)| + \sum_{v \in V^-} |v(c)|. \tag{3.7}$$

Next, replace the term $\sum_{v \in V^-} |v(c)|$ in (3.7) by an equivalent term that depends on $|p_a(c)|$ and $\sum_{v \in V^+} |v(c)|$. For this, use the following, derived from the definition of $p_a(c)$:

$$\begin{aligned} n \cdot p_a(c) &= \sum_{v \in V^+} |v(c)| - \sum_{v \in V^-} |v(c)| \\ \Leftrightarrow \sum_{v \in V^-} |v(c)| &= n \cdot (-p_a(c)) + \sum_{v \in V^+} |v(c)| \\ &= n \cdot |p_a(c)| + \sum_{v \in V^+} |v(c)|. \end{aligned}$$

The replacement results in

$$\sum_{v \in V} |l(c) - v(c)| = 2 \cdot \sum_{v \in V^+} |v(c)| + n \cdot |p_a(c)|$$
$$\geq n \cdot |p_a(c)| \geq n \cdot d.$$

This says that K-score$(V, C) \geq n \cdot d$, a contradiction to Claim 1.

Case 2: $l(c) \leq p_a(c) - d$.
Since $l(c) = 0$, the condition is equivalent to $0 \leq p_a(c) - d \Leftrightarrow d \leq p_a(c)$, and we have that $p_a(c)$ is nonnegative. Now, we show that Claim 2 also holds for this case.

$$\sum_{v \in V} |l(c) - v(c)| = \sum_{v \in V} |v(c)| = \sum_{v \in V^+} |v(c)| + \sum_{v \in V^-} |v(c)|$$
$$\geq \sum_{v \in V^+} v(c) + \sum_{v \in V^-} v(c) = p_a(c) \cdot n \geq d \cdot n.$$

Thus, also in this case, K-score$(V, C) \geq n \cdot d$, a contradiction to Claim 1.

□

Based on Lemma 3.2, for every position we can define the set of candidates that can take this position in an optimal Kemeny consensus. The subsequent definition will be useful for the formulation of the algorithm.

Definition 3.2. Let (V, C) be an election. For every integer $i \in \{0, \ldots, m-1\}$, let P_i denote the set of candidates that can assume the position i in an optimal Kemeny consensus, that is, $P_i := \{c \in C \mid p_a(c) - d < i < p_a(c) + d\}$.

Using Lemma 3.2, we can easily show the following.

Lemma 3.3. *For every position i, $|P_i| \leq 4d$.*

Proof. The proof is by contradiction. Assume that there is a position i with $|P_i| > 4d$. Due to Lemma 3.2, for every candidate $c \in P_i$ the positions which c may assume in an optimal Kemeny consensus can differ by at most $2d-1$. This is true because, otherwise, candidate c could not be in the given range around its average position. Then, in a Kemeny consensus, each of the at least $4d+1$ candidates must hold a position that differs at most by $2d-1$ from position i. As there are only $4d-1$ such positions ($2d-1$ on the left and $2d-1$ on the right of i), one obtains a contradiction. □

3.5.2 Dynamic programming algorithm

The dynamic programming algorithm for Kemeny Score exploits the fact that every candidate can only appear in a fixed range of positions in an optimal Kemeny consensus (see Lemma 3.2). The basic idea can be described as follows. The algorithm "generates" a Kemeny consensus from the left to the right. It tries out all possibilities for ordering the candidates locally and then combines these local solutions to yield an optimal Kemeny consensus. More specifically, according to Lemma 3.3, the number of candidates that can take a position i in an optimal Kemeny consensus for

any $i \in \{0, \ldots, m-1\}$ is at most $4d$. Thus, for position i, we can test all possible candidates. Having chosen a candidate for position i, the remaining candidates that could also assume i must either be left or right of i in a Kemeny consensus. Thus, we test all possible two-partitionings of this subset of candidates and compute a "partial" Kemeny score for every possibility. For the computation of the partial Kemeny scores at position i we make use of the partial solutions computed for the position $i-1$.

Definitions for the algorithm. For $i \in \{0, \ldots, m-1\}$, let $I(i)$ denote the set of candidates that could be "inserted" at position i *for the first time*, that is,

$$I(i) := \{c \in C \mid c \in P_i \text{ and } c \notin P_{i-1}\}.$$

Let $F(i)$ denote the set of candidates that must be "forgotten" at latest at position i, that is,

$$F(i) := \{c \in C \mid c \notin P_i \text{ and } c \in P_{i-1}\}.$$

For our algorithm, it is essential to subdivide the overall Kemeny score into *partial Kemeny scores* (pK). More precisely, for a candidate c and a subset R of candidates with $c \notin R$, we set

$$\mathrm{pK}(c, R) := \sum_{c' \in R} \sum_{v \in V} d_v^R(c, c'),$$

where for $c' \in R$ we have $d_v^R(c, c') := 0$ if in v we have $c > c'$, and $d_v^R(c, c') := 1$, otherwise. Intuitively, the partial Kemeny score denotes the score that is "induced" by candidate c and the candidate subset R if the candidates of R have higher positions than c in an optimal Kemeny consensus. Then, for a Kemeny consensus $l := c_0 > c_1 > \cdots > c_{m-1}$, the overall Kemeny score can be expressed by partial Kemeny scores as follows.

$$\text{K-score}(V, C) = \sum_{i=0}^{m-2} \sum_{j=i+1}^{m-1} \sum_{v \in V} d_{v,l}(c_i, c_j) \tag{3.8}$$

$$= \sum_{i=0}^{m-2} \sum_{c' \in R} \sum_{v \in V} d_v^R(c_i, c') \text{ with } R := \{c_j \mid i < j < m\} \tag{3.9}$$

$$= \sum_{i=0}^{m-2} \mathrm{pK}(c_i, \{c_j \mid i < j < m\}). \tag{3.10}$$

Next, consider the corresponding three-dimensional dynamic programming table T containing an entry for every position i, every candidate c that can assume i, and every candidate subset $C' \subseteq P_i \backslash \{c\}$. The entry stores the "minimum partial Kemeny score" over all possible orders of the candidates of C' under the condition that c takes position i and all candidates of C' take positions smaller than i. To define the dynamic programming table formally, we need some further notation.

Let $\Pi(C')$ denote the set of all possible orders of the candidates in C', where $C' \subseteq C$. Further, consider a Kemeny consensus in which every candidate of C' has a position smaller than every candidate in $C \backslash C'$. Then, the *minimum partial Kemeny score restricted to* C' is defined as

$$\min_{(d_1 > d_2 > \cdots > d_x) \in \Pi(C')} \left\{ \sum_{s=1}^{x} \mathrm{pK}\left(d_s, \{d_j \mid s < j < m\} \cup (C \backslash C')\right) \right\}$$

Input: An election (V, C) and, for every $0 \leq i < m$, the set P_i of candidates that can assume position i in an optimal Kemeny consensus.
Output: The Kemeny score of (V, C).

Initialization:
01 **for** $i = 0, \ldots, m-1$
02 **for all** $c \in P_i$
03 **for all** $P'_i \subseteq P_i \backslash \{c\}$
04 $T(i, c, P'_i) := +\infty$
05 **for all** $c \in P_0$
06 $T(0, c, \emptyset) := \mathrm{pK}(c, C \backslash \{c\})$

Update:
07 **for** $i = 1, \ldots, m-1$
08 **for all** $c \in P_i$
09 **for all** $P'_i \subseteq P_i \backslash \{c\}$
10 **if** $|P'_i \cup \bigcup_{j \leq i} F(j)| = i - 1$
and $T(i-1, c', (P'_i \cup F(i)) \backslash \{c'\})$ is defined **then**

11
$$T(i, c, P'_i) = \min_{c' \in P'_i \cup F(i)} T(i-1, c', (P'_i \cup F(i)) \backslash \{c'\}) + \mathrm{pK}(c, (P_i \cup \bigcup_{i<j<m} I(j)) \backslash (P'_i \cup \{c\}))$$

Output:
12 K-score $= \min_{c \in P_{m-1}} T(m-1, c, P_{m-1} \backslash \{c\})$

Figure 3.4: Dynamic programming algorithm for computing the Kemeny score of an election exploiting the average KT-distance as a parameter.

with $x := |C'|$. That is, it denotes the minimum partial Kemeny score over all orders of C'. We define an entry of the dynamic programming table T for a position i, a candidate $c \in P_i$, and a candidate subset $P'_i \subseteq P_i$ with $c \notin P'_i$. For this, we define $L := \bigcup_{j \leq i} F(j) \cup P'_i$. Then an entry $T(i, c, P'_i)$ denotes the minimum partial Kemeny score restricted to the candidates in $L \cup \{c\}$ under the assumptions that c is at position i in a Kemeny consensus, all candidates of L have positions smaller than i, and all other candidates have positions greater than i. That is, for $|L| = i - 1$, define

$$T(i, c, P'_i) := \min_{(d_1 > \cdots > d_{i-1}) \in \Pi(L)} \left\{ \sum_{s=0}^{i-1} \mathrm{pK}(d_s, C \backslash \{d_j \mid j \leq s\}) \right\} + \mathrm{pK}(c, C \backslash (L \cup \{c\})).$$

Algorithm. The dynamic programming algorithm is displayed in Figure 3.4. Its correctness and running time are proven in the following.

Lemma 3.4. *The algorithm in Figure 3.4 correctly computes* Kemeny Score.

Proof. We have to show two points:

First, all table entries are well-defined, that is, for an entry $T(i, c, P'_i)$ concerning position i there must be exactly $i-1$ candidates that have positions smaller than i. This condition is assured by line *10* of the algorithm.[1]

Second, we must ensure that our algorithm finds an optimal solution. Due to Equality (3.10), we know that the Kemeny score can be decomposed into partial Kemeny scores. Thus, it remains to show that the algorithm considers a decomposition that leads to an optimal solution. For every position, the algorithm tries all candidates in P_i. According to Lemma 3.2, one of these candidates must be the "correct" candidate c for this position. For c we can observe that the algorithm tries a sufficient number of possibilities to partition all remaining candidates $C \backslash \{c\}$ such that they have either smaller or greater positions than i. More precisely, every candidate from $C \backslash \{c\}$ must be in exactly one of the following three subsets:

1. The set F of candidates that have already been forgotten, that is, $F := \bigcup_{0 \leq j \leq i} F(j)$.
2. The set of candidates that can assume position i, that is, $P_i \backslash \{c\}$.
3. The set I of candidates that are not inserted yet, that is, $I := \bigcup_{i<j<m} I(j)$.

Due to Lemma 3.2 and the definition of $F(j)$, we know that a candidate from F cannot take a position greater than $i-1$ in an optimal Kemeny consensus. Thus, it is sufficient to explore only those partitions in which the candidates from F have positions smaller than i. Analogously, one can argue that for all candidates in I, it is sufficient to consider partitions in which they have positions greater than i. Thus, it remains to try all possibilities for partitioning the candidates from P_i. This is done in line *09* of the algorithm. Thus, the algorithm returns an optimal Kemeny score. □

Theorem 3.4. Kemeny Score *can be solved in* $O(16^d \cdot (d^2 \cdot m + d \cdot m^2 \log m \cdot n) + n^2 \cdot m \log m)$ *time with average KT-distance* d_a *and* $d = \lceil d_a \rceil$. *The size of the dynamic programming table is* $O(16^d \cdot d \cdot m)$.

Proof. The dynamic programming procedure requires the set of candidates P_i for $0 \leq i < m$ as input. To determine P_i for all $0 \leq i < m$, one needs the average positions of all candidates and the average KT-distance d_a of (V, C). To determine d_a, compute the pairwise distances of all pairs of votes. As there are $O(n^2)$ pairs and the pairwise KT-distance can be computed in $O(m \log m)$ time [145], this takes $O(n^2 \cdot m \log m)$ time. The average positions of all candidates can be computed in $O(n \cdot m)$ time by iterating once over every vote and adding the position of every candidate to a counter variable for this candidate. Thus, the input for the dynamic programming algorithm can be computed in $O(n^2 \cdot m \log m)$ time.

Concerning the dynamic programming algorithm itself, due to Lemma 3.3, for $0 \leq i < m$, the size of P_i is upper-bounded by $4d$. Then, for the initialization as well as for the update, the algorithm iterates over m positions, $4d$ candidates, and 2^{4d} subsets of candidates. Whereas the initialization in the innermost instruction (line *04*) can be

[1] It can still happen that a candidate takes a position outside of the required range around its average position. Since such an entry cannot lead to an optimal solution according to Lemma 3.2, this does not affect the correctness of the algorithm. To improve the running time it would be convenient to "cut away" such possibilities. We leave considerations in this direction to future work.

done in constant time, in every innermost instruction of the update phase (line *11*) one has to look for a minimum entry and one has to compute a pK-score. To find the minimum, one has to consider all candidates from $P'_i \cup F(i)$. As $P'_i \cup F(i)$ is a subset of P_{i-1}, it can contain at most $4d$ candidates. Furthermore, the required pK-score can be computed in $O(n \cdot m \log m)$ time. Thus, for the dynamic programming we arrive at the running time of $O(m \cdot 4d \cdot 2^{4d} \cdot (4d + n \cdot m \log m)) = O(16^d \cdot (d^2 \cdot m + d \cdot m^2 \log m \cdot n))$.

Concerning the size of the dynamic programming table, there are m positions and any position can be assumed by at most $4d$ candidates. The number of considered subsets is bounded from above by 2^{4d}. Hence, the size of the table T is $O(16^d \cdot d \cdot m)$. □

Finally, it is easy to modify the algorithm such that it outputs a Kemeny consensus: for every entry $T(i, c, P'_i)$, one additionally has to store a candidate c' that minimizes $T(i-1, c', (P'_i \cup F(i)) \setminus \{c'\})$ in line *11*. Then, starting with a minimum entry for position $m-1$, one can reconstruct an optimal Kemeny consensus by iteratively adding the "predecessor" candidate. The asymptotic running time remains unchanged.

3.6 Parameterizations by the candidate range

In this section, we consider two further parameterizations, namely "maximum range" and "average range" of candidates. While for the parameter "maximum range" we can obtain fixed-parameter tractability by using the dynamic programming algorithm described in Figure 3.4, Subsection 3.5.2, Kemeny Score becomes NP-complete already in case of an average range of two.

3.6.1 Maximum range

In the following, we show how to bound the number of candidates that can assume a position in an optimal Kemeny consensus by a function of the maximum range. This enables the application of the algorithm from Figure 3.4.

Lemma 3.5. *Let $r_{\max}$ be the maximum range of an election (V, C). Then, for every candidate its relative order in an optimal consensus with respect to all but at most $3r_{\max}$ candidates can be computed in $O(n \cdot m^2)$ time.*

Proof. According to Lemma 3.1, the following holds: If for two candidates $b, c \in C$ we have $v(b) > v(c)$ for all $v \in V$, then in every Kemeny consensus l it holds that $l(b) > l(c)$. Thus, it follows that for $b, c \in C$ with $\max_{v \in V} v(b) < \min_{v \in V} v(c)$, in an optimal Kemeny consensus l we have $l(b) < l(c)$. That is, for two candidates with "non-overlapping range" their relative order in an optimal Kemeny consensus can be determined using this observation. Clearly, all these candidate pairs can be computed in $O(n \cdot m^2)$ time.

Next, we show that for every candidate c there are at most $3r_{\max}$ candidates whose range overlaps with the range of c. The proof is by contradiction. Let the range of c contain the positions from i to j, with $i < j$. Assume that there is a subset of candidates $S \subseteq C$ with $|S| \geq 3r_{\max} + 1$ such that for every candidate $s \in S$ there is a vote $v \in V$ with $i \leq v(s) \leq j$. Now, consider an arbitrary input vote $v \in V$. Since there are at most $3r_{\max}$ positions p with $i - r_{\max} \leq p \leq j + r_{\max}$ for one candidate $s \in S$

it must hold that $v(s) < i - r_{\max}$ or $v(s) > j + r_{\max}$. Thus, the range of s is greater than $r_{\max}$, a contradiction. Hence, there can be at most $3r_{\max}$ candidates that have a position in the range of c in a vote $v \in V$. As described above, for all other candidates we can compute the relative order in $O(n \cdot m^2)$ time. Hence, the lemma follows. □

As a direct consequence of Lemma 3.5, we conclude that every candidate can assume one of at most $3r_{\max}$ consecutive positions in an optimal Kemeny consensus. Recall that for a position i the set of candidates that can assume i in an optimal consensus is denoted by P_i (see Definition 3.2). Then, using the same argument as in Lemma 3.3, one obtains the following.

Lemma 3.6. *For every position i, $|P_i| \leq 6r_{\max}$.*

In complete analogy to Theorem 3.4, one arrives at the following.

Theorem 3.5. Kemeny Score *can be solved in $O(32^{r_{\max}} \cdot (r_{\max}^2 \cdot m + r_{\max} \cdot m^2 \log m \cdot n) + n^2 \cdot m \log m)$ time with maximum range $r_{\max}$. The size of the dynamic programming table is $O(32^{r_{\max}} \cdot r_{\max} \cdot m)$.*

3.6.2 Average range

The following theorem shows that unless P=NP there is no fixed-parameter algorithm with respect to the "average range".

Theorem 3.6. Kemeny Score *is NP-complete for elections with average range two.*

Proof. The proof uses a many-one reduction from an arbitrary instance $((V, C), k)$ of Kemeny Score to a Kemeny Score-instance $((V', C'), k)$ with average range less than two. The construction of the election (V', C') is given in the following.

- $C' := C \uplus \{a_i \mid 1 \leq i \leq |C|^2\}$, that is, add $|C|^2$ new candidates.
- For every vote $v = c_1 > c_2 > \cdots > c_m$ in V, put the vote $v' := c_1 > c_2 > \cdots > c_m > a_1 > a_2 > \cdots > a_{m^2}$ into V'.

It follows from Lemma 3.1 that if a pair of candidates has the same order in all votes, it must have this order in a Kemeny consensus as well. Thus, in a Kemeny consensus it holds that $a_i > a_j$ for $i > j$ and, therefore, adding the candidates from $C' \backslash C$ does not increase the Kemeny score. Hence, an optimal Kemeny consensus of size k for (V', C') can be transformed into an optimal Kemeny consensus of size k for (V, C) by deleting the candidates from $C' \backslash C$. The average range of (V', C') is bounded as follows:

$$\begin{aligned} r_a &= \frac{1}{m + m^2} \cdot \sum_{c \in C'} r(c) \\ &= \frac{1}{m + m^2} \cdot \left(\sum_{c \in C} r(c) + \sum_{c \in C' \backslash C} r(c) \right) \\ &\leq \frac{1}{m + m^2} \cdot (m^2 + m^2) < 2. \end{aligned}$$

Clearly, the reduction can be easily modified to work for every constant value of at least two by choosing a C' of appropriate size. □

3.7 Ties and incomplete votes

In the following, we summarize results obtained for two generalizations of Kemeny Score [22]. See also Table 3.2, Section 3.8, for an overview. First, we allow for ties, that is, in a vote several candidates may be ranked equally. Second, we consider the scenario that one has only incomplete information.[2] Whereas most of our results can be transferred to the problem variant with ties, the Kemeny score problem behaves in a significantly different way in the case of incomplete information. Note that the algorithm from Section 3.3 regarding the parameter "number of candidates" directly applies to both generalizations.

3.7.1 Kemeny Score with Ties

A practically relevant extension of Kemeny Score is Kemeny Score with Ties [1, 132]. Here, one additionally allows the voters to classify sets of equally liked candidates, that is, a vote is no longer defined as a permutation of the candidates, but for two (or more) candidates a, b one can have $a = b$. The term $d_{v,w}(a, b)$ denoting the contribution of the candidate pair $\{a, b\}$ to the KT-distance between two votes v and w is modified as follows [132]. One has

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

In the literature there are different demands for the consensus itself. Hemaspaandra et al. [132] allow that the consensus list can contain ties as well. In contrast, Ailon [1] requires the consensus list to be a "full ranking", that is, a permutation of the candidates. We focus on the more general setting of Hemaspaandra et al. [132]. Note that Kemeny Score with Ties does not only generalize Kemeny Score but also includes other interesting special cases such as p-ratings and top-m lists [1].

Clearly, increasing the KT-distance by two if two votes strictly disagree on two candidates increases the overall score. Using analogous approaches to Section 3.4, one obtains a search tree of size at most 1.76^k and a kernelization result for the parameter "Kemeny score" [22, Theorem 7].

To deal with the structural parameterizations, we need to extend the notions of position and range. For a vote $v \in V$ and a candidate $c \in C$, let

$$T_v(c) := \{c' \in C \mid c \text{ and } c' \text{ tie in } v\}.$$

Now, for a vote v we can define the minimum and maximum position which c can assume. That is, $\mathrm{pos}_v^{\min}(c)$ is the number of candidates that are better than c in v and $\mathrm{pos}_v^{\max}(c) := \mathrm{pos}_v^{\min}(c) + |T_v(c)|$. Then, define the range of a candidate as

$$r(c) := \max_{v,w \in V} \{|\,\mathrm{pos}_v^{\max}(c) - \mathrm{pos}_w^{\min}(c)|\}.$$

The overall maximum range of an election is defined as the maximum of all candidate ranges. For Kemeny Score without ties, the algorithm used for showing fixed-parameter tractability for the average KT-distance strongly relies on the Lemma 3.2

[2] An alternative way to deal with incomplete information is provided by the Possible winner problem as introduced in Part II of this this thesis.

described in Subsection 3.5.1. However, it is not obvious how to transfer this result to the case with ties. The same is true for the parameterization by the maximum range: Here it is not obvious that Lemma 3.5 can be transferred to the case with ties. However, for both "maximum" parameters, we still can obtain fixed-parameter tractability by using a dynamic programming given for the parameterization by the "maximum KT-distance" for the case without ties in our conference paper [20]. This leads to the following. Kemeny Score with Ties can be solved in

$$O((3r_{\max}+1)! \cdot 2^{3r_{\max}+1} \cdot r_{\max} \log r_{\max} \cdot nm)$$

time with $r_{\max}$ being the maximum position range [22, Theorem 8] of a candidate and

$$O((6d_{\max}+2)! \cdot 2^{6d_{\max}+2} \cdot d_{\max} \cdot \log d_{\max} \cdot n \cdot m)$$

time with maximum KT-distance $d_{\max}$ [22, Theorem 9]. Regarding the "average KT-distance" we will obtain fixed-parameter tractability using a data reduction based framework described in Chapter 4 (see Theorem 4.1).

3.7.2 Kemeny Score with Incomplete Votes

The problem Kemeny Score with Incomplete Votes was introduced by Dwork et al. [77]. Here, the given votes are not required to be permutations of the entire candidate set, but only of candidate subsets, while the Kemeny consensus sought for should be a permutation of all candidates. In the definition of the KT-distance, set the pairwise distance between two votes to

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

where C_v contains the candidates occurring in vote v.

For incomplete votes we cannot apply the kernelization and the branching approach of Section 3.4 to show fixed-parameter tractability with respect to the Kemeny score. This is due to the fact that we can have non-trivial instances without dirty pairs. Using another approach based on a fixed-parameter algorithm for Feedback Arc Set [52], one can show that Kemeny Score with Incomplete Votes is solvable in $O(nm^2 + 2^k \cdot nm^2 + k!4^k \cdot m^4k^3)$ time [22, Theorem 10].

In incomplete votes the position of a candidate in a vote cannot be defined. Therefore, we cannot parameterize by the maximum range of candidate positions. By applying a simple many-one reduction from the NP-complete Feedback Arc Set problem, one can show that Kemeny Score with Incomplete Votes is NP-complete even if the maximum KT-distance between two input votes is zero. Hence, unless P=NP the problem cannot be fixed-parameter tractable with respect to the maximum as well as to the average KT-distance.

3.8 Further fixed-parameter algorithms

As mentioned in the corresponding sections, several of our results have been improved recently and some additional parameterizations have been investigated. The state of the art of the parameterized complexity of Kemeny Score is exhibited in Table 3.2 and the corresponding new results are briefly described in the following.

Table 3.2: Parameterized complexity of Kemeny Score and two of its generalizations. In case of positive results, we state the exponential parts of the corresponding running times if provided in the corresponding works. Results marked by (♣) follow from [77, 78], (♢) follow from [142], (♠) follow from [161], and (♡) follow from [25] and are discussed in Chapter 4. The remaining results are provided in this chapter.

	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	$2^{O(\sqrt{k})}$ (♢)	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	$2^{O(\sqrt{d_m})}$ (♢)	$(6d_m + 2)! \cdot 2^{6d_m+2}$	NP-h for $d_m = 0$
avg. KT-dist d_a	$2^{O(\sqrt{d_a})}$ (♢)	$2^{O(d_a^2)}$ (♡)	NP-h for $d_a = 0$
$\overline{d} := k/n$	$2^{O(\sqrt{\overline{d}})}$ (♢)	$2^{O(\overline{d}^2)}$ (♡)	NP-h for $\overline{d} = 0$
above guarantee	FPT (♠)	?	?

Average distance from a Kemeny consensus. Simjour [189] introduced the parameter average distance $\overline{d}$ from a Kemeny consensus to the input votes defined by

$$\overline{d} := 1/n \cdot \sum_{v \in V} \mathrm{dist}(v, l) = k/n$$

for a Kemeny consensus l with Kemeny score k.[3] As pointed out by Simjour and also used by Karpinski and Schudy [142], the average KT-distance d_a is clearly at least $\overline{d}$. In the following, we show that d_a is at most twice as large as $\overline{d}$ and hence both parameters are "equivalent" up to the factor two. Since the KT-distance is a metric, by the triangle inequality the following is easy to see:

$$\begin{aligned} d_a &= \sum_{v \in V} \sum_{w \in V \setminus \{v\}} \mathrm{dist}(v, w)/(n(n-1)) \\ &\leq \sum_{v \in V} \sum_{w \in V \setminus \{v\}} \big(\mathrm{dist}(v, l) + \mathrm{dist}(l, w)\big)/(n(n-1)) \\ &= 2 \cdot \big(\sum_{v \in V} \sum_{w \in V \setminus \{v\}} \mathrm{dist}(v, l)\big)/(n(n-1)) \\ &= 2 \cdot \big(\sum_{v \in V} \mathrm{dist}(v, l)\big)/n = 2\overline{d}. \end{aligned}$$

Hence, we arrive at the following.

[3] An analogous parameter for the NP-hard Consensus Pattern problem has been considered by Marx [162].

Observation 3.4. *For every election it holds that* $d_a \geq \overline{d} \geq d_a/2$.

Note that it directly follows that Observations 3.1, 3.2, and 3.3 stating the "independence" from the range to the distance parameterizations carry over to $\overline{d}$.

Simjour improved the running times provided in this chapter for the parameters k, d_m, d_a, and $\overline{d}$ by decreasing the constant of the basis. The underlying algorithmic ideas rely on a transformation of Kemeny Score to Weighted Feedback Arc Set and applying and extending known algorithms for Weighted Feedback Arc Set. Recently, Karpinski and Schudy [142] provided a fixed-parameter algorithm with respect to $\overline{d}$ having a subexponential running time in $\overline{d}$. Clearly, this implies subexponential running times with respect to k, d_m, and d_a as well.

Above guaranteed value. Parameterization above guaranteed value has been introduced by Mahajan and Raman [160]. For Kemeny Score, the point is that

$$L := \sum_{\{a,b\} \subseteq C} \min\{\pi(a,b), \pi(b,a)\},$$

where $\pi(a,b)$ denotes the number of votes that rank a higher than b, is an obvious lower bound for the Kemeny Score k. Hence, it is interesting to parameterize above this guaranteed lower bound, more precisely, by the parameter "$k-L$". Applying a parameter preserving-reduction from Kemeny Score to a weighted variant of Directed Feedback Vertex Set, Mahajan et al.[161] observed fixed-parameter tractability with respect to $k-L$.

3.9 Conclusion

We initiated a multivariate complexity analysis of Kemeny Score including the identification of meaningful parameterizations such as the "average KT-distance" and "candidate range". Our corresponding results are displayed in Table 3.1 (Section 3.2). In the meantime our results have been extended and some of them improved by several authors [142, 161, 189]. An overview of the state of the art of the parameterized complexity of Kemeny Score and the two generalizations allowing for ties or incomplete information is given Table 3.2. There are numerous challenges for future studies:

- In several applications, it is useful to compute not just *one* optimal Kemeny consensus but to enumerate all of them. Simjour [189] provided a search-based algorithm for enumerating all Kemeny consensuses and showed that the exponential part of the running time is at most $36^{\overline{d}}$. Can this result be improved and extended to some other parameterizations?

- A long-standing open question regards the computational complexity of Kemeny Score with three votes [77, 2]. From the many-one reduction provided by Dwork et al. [77], NP-hardness follows only for an *even* number of votes, that is, for all odd fixed values such as 3,5,7,... the computational complexity is still open.

- Regarding Kemeny Score with Ties the running times for many parameters as provided in Table 3.2 still have a high combinatorial explosion and hence seek

for improvement. Note that the results for the "average KT-distance" rely on an approach based on data reduction (see Chapter 4). Some additional non-data reduction based algorithm complementing this result would be desirable. While the strategy provided in this chapter does not immediately transfer to the case with ties (see Subsection 3.7.1), it might be possible to adapt the algorithms provided by Simjour [189].

It might be also interesting to investigate whether special cases of Kemeny Score with Ties such p-ratings and top-m lists [1] allow for more efficient algorithms.

In addition to the above questions regarding Kemeny Score directly, there are several closely related problems for which it might be interesting to investigate how far the results obtained for Kemeny Score can be transfered.

- Kemeny Score is a median problem seeking to minimize the *sum* of distances from a preference list. Analogously, one can seek for a preference list minimizing the *maximum* distance (that is, searching for the center instead of the median). Due to an application in graph drawing, this problem is known as Crossing Permutation and its computational complexity has been investigated by Biedl et al. [32]. Some first results regarding the parameterized complexity have been obtained by Schwarz [186]. One possible interpretation of Crossing Permutation in the context of voting concerns scenarios in which it is mandatory to protect minorities. Then, one might look for an outcome of an election that minimizes the damage for the "most aggrieved voter".

- Fagin et al. [92] introduced various distance measures between "top k lists", for example, the Hausdorff Kendall distance. For every such distance between two lists one can define a consensus problem analogous to Kemeny Score.

- The Metric s-Median problem can be stated as follows (see Shindler [188] for a survey). Given a set N of points in some metric space and some integers s and k, it asks whether there is a size-s subset $K \subset N$ such that the sum of all N's points' distances to their nearest element of K is at most k. Since the KT-distance is a metric, Kemeny Score can be considered as a special case of this problem with $s = 1$, that is, searching for one consensus. It might be interesting to identify scenarios where one is looking for a set of consensus ranking and investigate the computational complexity of the corresponding problems. From a voting point of view this directly leads to a "multiple winner" scenario.

In the following chapter, we further extend our results for the parameter average KT-distance by developing a new data reduction methodology.

Chapter 4

Partial kernelization for Kemeny

In the previous chapter, we started a multivariate analysis of Kemeny Score which will be further extended in this chapter. We provide some polynomial-time data reduction rules with performance guarantee for Kemeny Score. More specifically, we show that the number of candidates in a reduced instance only depends on the "average KT-distance" and another, newly introduced parameter. Then, fixed-parameter tractability with respect to these parameters follows from the fixed-parameter algorithm with respect to the "number of candidates" from Section 3.3. Although for the parameter "average KT-distance" d_a our results do not improve the bound on the worst-case running time of $2^{O(\sqrt{d_a})} \cdot \text{poly}$ [142] (see also Table 7.1), efficient polynomial-time data reduction clearly complements the previous results. Experiments showing the practical value of data reduction for the computation of Kemeny rankings are provided in Chapter 5.

Methodology. The results of this chapter rely on a new methodological framework for intractable median problems such as Kemeny Score. In median problems one is given a set of objects and the task is to find a "consensus object" that minimizes the sum of distances to the given input objects. The framework was exhibited for Kemeny Score with and without ties, and the problems Swap Median Permutation and Consensus Clustering [25]. Here, we only focus on Kemeny Score. Our algorithmic framework shows that if the input objects are sufficiently "similar on average", then there are provably effective data reduction rules.

Within our framework, two points deserve particular attention. First, the identification of *polynomial-time solvable special cases* of the underlying problems. Second, a novel concept of kernelization based on polynomial-time data reduction that does not yield problem kernels in the classical sense of parameterized algorithmics but still allows for "partial problem kernels". The basic idea can be explained as follows. In multi-dimensional problems, a partial kernelization reduces at least one dimension such that its size only depends on the parameter value. In our case, the reduced dimension refers to the "number of candidates" and the parameter to the "average KT-distance". The concept of partial kernelization promises to be useful beyond the problems and parameterizations for which the framework has been exhibited [25].

On the way to proving our results with respect to the parameter "average distance", we introduce another measurement of dissimilarity—the "number of dirty elements"—which can be considered as an alternative parameterization. We also show fixed-parameter tractability with respect to this parameterization. As we will see, both parameterizations are closely related. In comparison, the "average distance" seems to be the more intuitive and easier to understand parameter whereas the "dirtiness" parameterization seems to yield stronger results.

Results. Our results for Kemeny Score are summarized as follows. We introduce a concept of "dirtiness" for candidates and pairs of candidates. This concept is used to identify a polynomial-time solvable special case and allows for efficient data reduction rules resulting in a linear-size partial kernel with respect to the "average KT-distance" d_a. More specifically, our new data reduction rules can transform every instance into an equivalent one that contains less than $11d_a$ candidates. We further classify different "degrees of dirtiness" and, depending on this degree, either obtain a linear or quadratic partial kernel with respect to the "number of dirty pairs". Finally, we briefly discuss analogous results for Kemeny Score with Ties settling the so far open question of fixed-parameter tractability with respect to d_a.

4.1 Framework and basic definitions

The outline of our framework adapted to Kemeny Score reads as follows.

Step 1. Identify a polynomial-time solvable special case by defining a concept of "dirtiness" for candidates and proving that an instance without dirty candidates can be easily solved.

Step 2. Show that the number of dirty candidates is bounded from above by a polynomial only depending on the average KT-distance.

Step 3. Develop polynomial-time data reduction rules such that in a reduced instance the number of nondirty candidates is bounded from above by a polynomial only depending on the number of dirty candidates and, thus, also only depending on the average distance.

Step 4. Exploit the fact that Kemeny Score is fixed-parameter tractable with respect to the number of candidates (see Section 8.1).

This framework yields fixed-parameter tractability with respect to both parameters "average KT-distance" and "number of dirty candidates". In general, fixed-parameter tractability would also follow for nonpolynomial functions in Steps 2 and 3, but all our results provide polynomial bounds. A special feature of our framework is that in Step 3 we perform a "partial kernelization", a concept of general interest. Herein, the term "partial" refers to the fact that only the number of candidates is reduced, but not the number of votes. This leads to the following general definition.

Definition 4.1. Let (I, k) be an instance of a parameterized problem P, where $I \in \Sigma^*$ denotes the input instance and k a parameter. Let $d : \Sigma^* \to \mathbb{N}$ be a computable function such that P is fixed-parameter tractable with respect to $d(I)$. The problem

admits a *partial kernel* if there is a polynomial-time algorithm that computes an instance (I', k') of P such that:

- (I, k) is a yes-instance if and only if (I', k') is a yes-instance,
- $k' \leq f(k)$, and
- $d(I') \leq g(k)$

for computable functions f and g.

For I, k, and d meeting the above conditions, the existence of a partial kernel directly implies fixed-parameter tractability with respect to the parameter k. Our partial kernelization can be seen as a generalization of "classical" problem kernelization that reduces an instance of a problem to an instance whose size is bounded by a function of the parameter: Choosing $d(I) := |I|$ directly leads to the classical problem kernel definition.

Basic definitions regarding Kemeny Score and several parameterizations are provided in the introduction of Chapter 3. To show the following results, it will be useful to decompose the Kemeny score of a preference list into "partial scores". More precisely, for a preference list l and a candidate pair $\{a, b\}$, the *partial score* of l with respect to $\{a, b\}$ is

$$s_l(\{a, b\}) := \sum_{v \in V} d_{v,l}(a, b).$$

The partial score of l with respect to a subset P of candidate pairs is $s_l(P) := \sum_{p \in P} s_l(p)$. The following notation will be useful to state some of our reduction rules. For a candidate subset $C' \subseteq C$, we say that a ranking fulfills the condition $C' > C \setminus C'$ if every candidate from C' is preferred to every candidate from $C \setminus C'$.

4.2 Dirtiness and a polynomial-time special case

We measure the dirtiness of a pair of candidates by the amount of agreement of the votes for this pair. To this end, we introduce the following notation.

Definition 4.2. For an election (V, C), two candidates $c, c' \in C$, and a rational number $s \in]0.5, 1]$, we write

$$c \geq_s c'$$

if at least $\lceil s \cdot |V| \rceil$ of the votes prefer c to c'. A candidate pair $\{c, c'\}$ is *dirty according to the $\geq_s$-majority* if neither $c \geq_s c'$ nor $c' \geq_s c$. All remaining pairs are *nondirty according to the $\geq_s$-majority.*

We say that c and c' are *ordered according to the $\geq_s$-majority* in a preference list l if $c \geq_s c'$ and $c > c'$ in l. In the following, we show that if all candidate pairs are nondirty with respect to the $\geq_s$-majority for an $s > 2/3$, then there exists a *$\geq_s$-majority order*, that is, a preference list in which all candidate pairs are ordered according to the $\geq_s$-majority. To simplify matters, we write "$>_{2/3}$" instead of "$\geq_s$ with $s > 2/3$", and if the value of s is clear from the context, then we just speak of "dirty pairs" and omit "according to the $\geq_s$-majority". Candidates appearing only in

$$a > b > c$$
$$b > c > a$$
$$c > a > b$$

Figure 4.1: Example for the nonexistence of a $\geq_s$-majority list with $s \leq 2/3$. Since $a \geq_s b$, $b \geq_s a$, and $c \geq_s a$ for any $s \in]0.5, 2/3]$, there is no linear order fulfilling all three relative orders.

nondirty pairs are called *nondirty candidates* and all remaining candidates are *dirty candidates*. Note that with this definition a nondirty pair can also be formed by two dirty candidates. The number of dirty candidates is closely related to the number of dirty pairs. More specifically, x dirty candidates can form at most $\binom{x}{2}$ dirty pairs and x dirty pairs consist of at most $2x$ dirty candidates. Hence, fixed-parameter tractability for one parameter implies fixed-parameter tractability with respect to the other parameter as well. In the following, we mainly focus on the parameterization by the "number of dirty pairs".

Polynomial-time special case. Now, we show how a limited amount of dirtiness allows for a polynomial-time solvable special case as required for the first step of our framework.

Proposition 4.1. *For any $s \in]2/3, 1]$, a* Kemeny Score *instance without dirty pairs according to the $\geq_s$-majority can be decided in polynomial time. The unique Kemeny consensus is provided by the $\geq_s$-majority order.*

Proof. For an input instance (V, C, k) of Kemeny Score without dirty pairs according to the $\geq_s$-majority with $s > 2/3$, we show that the preference list "induced" by the $>_{2/3}$-majority of the candidate pairs is optimal.

First, we show by contradiction that there is a preference list $l_{2/3}$ where for all candidate pairs $\{a, b\}$ with $a, b \in C$ and $a >_{2/3} b$, one has $a > b$. Assume that such a preference list does not exist. Then, there must be three candidates $a, b, c \in C$ violating transitivity, that is, $a >_{2/3} b$, $b >_{2/3} c$, and $c >_{2/3} a$. Since $a >_{2/3} b$ and $b >_{2/3} c$, there must be at least $n/3$ votes with $a > b > c$. Since a and c do not form a dirty pair, it follows that $a >_{2/3} c$, a contradiction.

Second, we show by contradiction that $l_{2/3}$ is optimal. Assume that there is a Kemeny consensus l with a nonempty set P of candidate pairs that are not ordered according to the $>_{2/3}$-majority; that is, $P := \{\{c, c'\} : c > c' \text{ in } l \text{ and } c' >_{2/3} c\}$. All candidate pairs that are not in P are ordered equally in l and $l_{2/3}$. Thus, the partial score with respect to them is the same for l and $l_{2/3}$. For every candidate pair $\{c, c'\} \in P$, the partial score $s_l(\{c, c'\})$ is more than $2n/3$ and the partial score $s_{l_{2/3}}(\{c, c'\})$ is less than $n/3$. Thus, the score of $l_{2/3}$ is smaller than the score of l, a contradiction to the optimality of l. □

Complementing Proposition 4.1, the example provided in Figure 4.1 shows that for any $s \leq 2/3$, a $\geq_s$-majority order does not necessarily exist.

Bound on the number of dirty pairs. Following Step 2 of our framework, the next lemma shows how the number of dirty pairs is bounded from above by a function linear in the average KT-distance d_a.

Table 4.1: Overview of properties induced by $\geq_s$-majorities for different values of s.

value of s	induced properties
$1/2 \leq s < 2/3$	a $\geq_s$-majority order does not exist (Example 4.1)
$2/3 < s < 3/4$	a $\geq_s$-majority order exists (Proposition 4.1) **but** a nondirty candidate and a dirty candidate have not to be ordered according the $\geq_s$-majority in a Kemeny consensus (Theorem 4.2)
$3/4 \leq s \leq 1$	a $\geq_s$-majority order exists (Proposition 4.1) **and** in every Kemeny consensus every nondirty candidate is ordered according to the $\geq_s$-majority wrt. every other candidate (Lemma 4.2)

Lemma 4.1. *For any fixed rational s in $]2/3, 1]$, for an instance of* Kemeny Score *with average KT-distance d_a, there are less than $\gamma \cdot d_a$ dirty pairs according to the $\geq_s$-majority for an appropriate constant γ.*

Proof. For an election (V, C) with average KT-distance d_a, let i denote the number of dirty pairs. Every dirty pair $\{a, b\} \subseteq C$ contributes at least

$$sn \cdot (1-s)n$$

to the overall sum of KT-distances. Recall that

$$d_a = \left(\sum_{v,w \in V} \mathrm{dist}(v, w) \right) / (n(n-1)) = \left(\sum_{\{c,d\} \subseteq C} \sum_{v,w \in V} d_{v,w}(c, d) \right) / (n(n-1)).$$

Thus,

$$d_a \geq \frac{1}{n(n-1)} \cdot i \cdot s \cdot (1-s) \cdot n^2 > s(1-s) \cdot i \;\Leftrightarrow\; \frac{1}{s \cdot (1-s)} \cdot d > i.$$

Hence, γ can be set to $1/(s(1-s))$ which is constant for fixed s. □

The constant γ in Lemma 4.1 strongly depends on the value of s, that is, on the kind of dirtiness. More specifically,

$$\gamma = \frac{1}{s(1-s)}.$$

Hence, for increasing values of s also the upper bound on the number of dirty pairs increases. Intuitively, this makes sense since for an arbitrary election having a fixed average KT-distance, the stronger the requirements for a *nondirty* pair are, the more *dirty* pairs must exist. For relevant values of s, the constant γ is of reasonable size. For $s > 2/3$, one has $\gamma \geq 9/2$, and, for example, for $s = 3/4$, one still obtains $\gamma = 16/3$.

4.3 Data reduction rules and partial kernelization

The previous section provided a concept of dirtiness and showed that the number of dirty pairs is bounded from above by a function linear in the average KT-distance.

In this section, following the third step of our framework, we develop data reduction rules aiming at the deletion of nondirty candidates. As we will see in the following, for different values of s, we obtain different "qualities" of data reduction. More specifically, whereas we provide a linear partial kernel for $s \geq 3/4$, we only provide a partial kernel of quadratic size for $2/3 < s < 3/4$; in both cases with respect to to the "average KT-distance" as well as with respect to the "number of dirty pairs". We also give evidence that the approach provided for $s \geq 3/4$ cannot be transferred to any $s < 3/4$.

The different results for different values of s rely on some structural properties "induced" by the $\geq_s$-majority as provided in Table 4.1. In particular, we will show in the following that for $s \geq 3/4$, in every Kemeny consensus every nondirty candidate must be ordered according to the $\geq_s$-majority with respect to every other candidate. In contrast, for smaller values of s it may happen that there is a nondirty pair formed by a dirty and a nondirty candidate such that there is no Kemeny consensus in which this pair is ordered according to the $\geq_s$-majority order.

4.3.1 Exploiting $\geq_{3/4}$-majorities

In this subsection, we provide a data reduction rule depending on the $\geq_{3/4}$-majority. We show that this reduction rules leads to a linear partial kernel with respect to the "average KT-distance" as well as with respect to the "number of dirty candidates" and the "number of dirty pairs", respectively. Clearly, all results also hold for any $\geq_s$-majority with $s \geq 3/4$. The following lemma allows us to formulate a data reduction rule that deletes all nondirty candidates and additionally may break the remaining set of dirty candidates into several subsets to be handled independently from each other.

Lemma 4.2. *Let $a \in C$ be a nondirty candidate with respect to the $\geq_{3/4}$-majority and $b \in C \setminus \{a\}$. If $a \geq_{3/4} b$, then in every Kemeny consensus one must have "$a > \cdots > b$"; if $b \geq_{3/4} a$, then in every Kemeny consensus one must have "$b > \cdots > a$".*

Proof. We consider the case $a \geq_{3/4} b$; the case $b \geq_{3/4} a$ follows in complete analogy. The proof is by contradiction. Assume that there is a Kemeny consensus l with "$\ldots > b > D > a > \ldots$" for some $D \subseteq C \setminus \{a, b\}$. Since a is nondirty, for every candidate $d \in D$, either $a \geq_{3/4} d$ or $d \geq_{3/4} a$. Let $D_1 := \{d \in D \mid a \geq_{3/4} d\}$ and $D_2 := \{d \in D \mid d \geq_{3/4} a\}$. Consider the preference list l' obtained from l by replacing

$$b > D > a$$

by

$$D_2 > a > b > D_1,$$

where the positions of all other candidates remain unchanged and the candidates within D_1 and D_2 have the same relative order as within D. We show that the score of l is greater than the score of l' contradicting that l is a Kemeny consensus. The only candidate pairs that have different orders in l and l' and thus the only candidate pairs that can contribute with different partial scores to the scores of l and l' are $\{a, d\}$ and $\{d_2, d\}$ for all $d \in D_1 \cup \{b\}$ and all $d_2 \in D_2$. Consider any $d \in D_1 \cup \{b\}$ and $d_2 \in D_2$. Since $|\{v \in V \mid d_2 \geq_{3/4} a\}| \geq 3/4 \cdot |V|$ and $|\{v \in V \mid a \geq_{3/4} d\}| \geq 3/4 \cdot |V|$, the intersection of these two sets must contain at least $|V|/2$ elements, that is, there must be at least $|V|/2$ votes with "$d_2 > \cdots > a > \cdots > d$". Thus, the partial score of $\{d_2, d\}$ in l is at least as high as its partial score in l'. The partial score of every pair $\{a, d\}$

with $d \in D_1 \cup \{b\}$ in l' is strictly less than the partial score in l. Since $|D_1 \cup \{b\}| \geq 1$, the score of l' is smaller than the score of l and thus l cannot be a Kemeny consensus, a contradiction. □

As a direct consequence of Lemma 4.2 we can partition the candidates of an election (V, C) as follows. Let $N := \{n_1, \ldots, n_s\}$ denote the set of nondirty candidates with respect to the $\geq_{3/4}$-majority such that $n_i \geq_{3/4} n_{i+1}$ for $1 \leq i \leq s-1$. Then, $D_0 := \{d \in C \setminus N \mid d \geq_{3/4} n_1\}$,

$$D_i := \{d \in C \setminus N \mid n_i \geq_{3/4} d \text{ and } d \geq_{3/4} n_{i+1}\} \text{ for } 1 \leq i \leq s-1,$$

and $D_s := \{d \in C \setminus N \mid n_s \geq_{3/4} d\}$. Furthermore, a subinstance of (V, C) *induced* by a candidate subset $C' \subseteq C$ is given by (V', C') where every vote in V' one-to-one corresponds to a vote in V keeping the relative order of the candidates from C'.

Rule 4.1. (**3/4-Majority Rule**[1]) Let (V, C) be an election and N and $D_0, \ldots, D_s$ be the sets of nondirty and dirty candidates as specified above. Delete (V, C) and keep the $s+1$ subinstances induced by D_i for $i \in \{0, \ldots, s\}$.

The soundness of the 3/4-Majority Rule follows directly from Lemma 4.2 and it is straightforward to verify that it runs in $O(nm^2)$ time. It is easy to adapt the 3/4-Majority Rule to work for the decision problem: An instance is reduced by deleting all candidates from N, reordering every vote such that $D_0 > D_1 > \cdots > D_s$ where within D_i, $0 \leq i \leq s$, the order of the candidates remains unchanged, and decreasing the Kemeny score appropriately. Making use of a simple relation between the number of dirty candidates and the average KT-distance, one arrives at the following.

Theorem 4.1. Kemeny Score *admits a partial kernel with less than* $11 \cdot d_a$ *candidates where* d_a *denotes the average KT-distance and all candidates of the partial kernel are dirty according to the* $\geq_{3/4}$*-majority. The partial kernel can be computed in* $O(nm^2)$ *time for an election with* n *votes and* m *candidates.*

Proof. After applying the 3/4-Majority Rule, only dirty candidates remain. Let their number be i. Since every dirty candidate must be involved in at least one candidate pair that is not ordered according to the $\geq_{3/4}$-majority, there must be at least $i/2$ candidate pairs that contribute more than $n/4 \cdot 3n/4$ to the average KT-distance of the original input instance. By definition of the average KT-distance, it follows that

$$d_a > \frac{1}{n(n-1)} \cdot \frac{i}{2} \cdot \frac{n}{4} \cdot \frac{3n}{4} > \frac{3}{32} \cdot i \Rightarrow 11 \cdot d_a > i.$$

□

4.3.2 Tightness of the 3/4-Majority Rule

The existence of a reduction rule analogously to the 3/4-Majority Rule for $\geq_s$-majorities for $s < 3/4$ would be desirable since such a rule might be more effective: There are instances for which a candidate is dirty according to the $\geq_{3/4}$-majority but nondirty according to a $\geq_s$-majority with $s < 3/4$. In the following, we discuss why such a

[1] This reduction rule will play an important rule in the following chapter.

reduction rule cannot exist. The decisive point of the 3/4-Majority Rule is that, in a Kemeny consensus, every nondirty candidate must be ordered according to the $\geq_{3/4}$-majority with respect to *every* other candidate. The following theorem shows that this is not true for $\geq_s$-majorities with $s < 3/4$:

Theorem 4.2. *Consider the $\geq_s$-majority for any rational $s \in]2/3, 3/4[$. Then, for a nondirty candidate x and a dirty candidate y, $x \geq_s y$ does not imply $x > y$ in a Kemeny consensus.*

Proof. Let s_1 and s_2 be two positive integers such that $s = s_1/s_2$. We construct an election such that there is a nondirty candidate x with $x \geq_s y$ but "$y > \cdots > x$" in every Kemeny consensus. The set of candidates is $\{x, y, a_1, a_2\}$ and there are the following $n = s_1 \cdot s_2$ votes:

- $s_1 \cdot s_2 - s_1^2$ votes of type 1: $x > y > a_1 > a_2$
- $2s_1^2 - s_1 \cdot s_2$ votes of type 2: $a_1 > a_2 > x > y$
- $s_1 \cdot s_2 - s_1^2$ votes of type 3: $y > a_1 > a_2 > x$

We first show that the votes are well-defined, that is, there is a positive number of votes of every type and the total number of votes is $s_1 \cdot s_2$:

The total number of votes is

$$s_1 \cdot s_2 - s_1^2 + 2s_1^2 - s_1 \cdot s_2 + s_1 \cdot s_2 - s_1^2 = s_1 \cdot s_2.$$

Considering the number of votes of types 1 and 3, recall that $3/4 > s_1/s_2$ and thus $s_2 > 4/3 \cdot s_1$. Hence, it is easy to see that their number is

$$s_1 \cdot s_2 - s_1^2 > s_1 \cdot (4/3 \cdot s_1 - s_1) > 0.$$

Regarding votes of type 2, we use the trivial bound that $s_1/s_2 > 1/2$ and thus their number is

$$2s_1^2 - s_1 \cdot s_2 > s_1 \cdot (2s_1 - 2s_1) = 0.$$

In the remainder of the proof, we show the following two points:

1. x is nondirty and $x \geq_s y$.
2. The score of "$y > a_1 > a_2 > x$" is smaller than the score of every other preference list.

(1.): The number of votes with $a > x$ for $a \in \{a_1, a_2\}$ is

$$2s_1^2 - s_1 \cdot s_2 + s_1 \cdot s_2 - s_1^2 = s_1^2 = s \cdot n$$

and the number of votes with $x > y$ is

$$s_1 \cdot s_2 - s_1^2 + 2s_1^2 - s_1 \cdot s_2 = s_1^2 = s \cdot n$$

and thus x is nondirty according to the $\geq_s$-majority and $x \geq_s y$.

(2.): Due to the Extended Condorcet criterion [191], $a_1 > a_2$ in every Kemeny consensus. Distinguishing three cases, we first show that in every Kemeny consensus $a_1 > x$ if and only if $a_2 > x$, and $a_1 > y$ if and only if $a_2 > y$. After this, we can treat a_1 and a_2 as one candidate of "weight" two and thus with this argument there remain only six preference lists for which the score has to be investigated to show that "$y > a_1 > a_2 > x$" is the only preference list with minimum score.

- Case 1: Consider a preference list with "$a_1 > x > a_2$" where y is placed either before or after the other three candidates. This preference list cannot have minimum score since swapping x and a_2 leads to a preference list with smaller score (since $a_2 \geq x$ in more than $sn > 2/3 \cdot n$ votes).

- Case 2: Consider a preference list with "$a_1 > y > a_2$" where x is placed either before or after the three other candidates. This preference list cannot have minimum score swapping a_1 and y leads to a preference list with smaller score which can be seen as follows. Since $s_1 < 3/4 \cdot s_2$, the number of votes with $y > a_1$ is
$$2s_1s_2 - 2s_1^2 > 2s_1(s_2 - 3/4 \cdot s_2) = 1/2 \cdot s_1s_2 = n/2.$$

- Case 3: Consider the preference list "$a_1 > x > y > a_2$". (Clearly, the same preference list with x and y swapped would have a larger score.) We show that $a_1 > a_2 > x > y$ has a smaller score. The only pairs that change the score are $\{a_2, y\}$ and $\{a_2, x\}$. These pairs contribute with
$$\#_v(a_2 > y) + \#_v(a_2 > x) = 2s_1^2 - s_1s_2 + 2s_1^2 - s_1s_2 + s_1s_2 - s_1^2 = 3s_1^2 - s_1s_2$$
to the old score and with $2n - \#_v(a_2 > y) - \#_v(a_2 > x)$ to the "new" score. Hence, it remains to show that the difference between the old and new score is positive, that is,
$$3s_1^2 - s_1s_2 - 2s_1s_2 + 3s_1^2 - s_1s_2 = 6s_1^2 - 4s_1s_2 > 6 \cdot 2/3 \cdot s_1s_2 - 4s_1s_2 = 0.$$

Finally, we consider the scores of all possible remaining six preference lists $r_1, \ldots, r_6$ with a standing for "$a_1 > a_2$":

$$\begin{array}{llll} r_1: & a > x > y & r_4: & x > y > a \\ r_2: & a > y > x & r_5: & y > a > x \\ r_3: & x > a > y & r_6: & y > x > a \end{array}$$

Let $t(r)$ denote the score of a preference list r. It is easy to verify that $t(r_1) < t(r_2)$ and $t(r_1) < t(r_3)$ and that $t(r_4) < t(r_6)$. Hence, it is sufficient to compare the score of r_5 with the score of r_1 and r_4. Since a represents two candidates, we count the corresponding pairs twice in the following computations.

$$\begin{aligned} &t(r_1) - t(r_5) \\ &= 2\#_v(x > a) + 2\#_v(y > a) + \#_v(y > x) - 2\#_v(a > y) - 2\#_v(x > a) - \#_v(x > y) \\ &= 2s_1s_2 - 2s_1^2 + 4s_1s_2 - 4s_1^2 + s_1s_2 - s_1^2 - 4s_1^2 + 2s_1s_2 - 2s_1s_2 + 2s_1^2 - s_1^2 \\ &= 7s_1s_2 - 6s_1^2 > 7s_1 \cdot 4/3 \cdot s_1 - 6s_1^2 = 10/3 \cdot s_1^2 > 0 \end{aligned}$$

$$\begin{aligned} &t(r_4) - t(r_5) \\ &= \#_v(y > x) + 2\#_v(a > x) + 2\#_v(a > y) - 2\#_v(a > y) - 2\#_v(x > a) - \#_v(x > y) \\ &= s_1s_2 - s_1^2 + 2 \cdot s_1^2 - 2 \cdot (s_1s_2) + 2 \cdot s_1^2 - s_1^2 \\ &= 2s_1^2 - s_1s_2 > 2s_1^2 - 3/2 \cdot s_1^2 = 1/2 \cdot s_1^2 > 0 \end{aligned}$$

Altogether, we showed that r_5 is the only Kemeny consensus. Thus, there is an election with $x \geq_s y$ for every $s \in]2/3, 3/4[$ such that every Kemeny ranking has $y > x$. □

Note that in this construction of the counterexample, the number of votes can become quite high, for example, for values of s which are very close to 3/4. However, for such cases it is also possible to construct a counterexample with a smaller number of votes by choosing an (appropriate) smaller number of $n = s'$ votes such that $\lceil s'n \rceil = \lceil sn \rceil$. We omit the details.

4.3.3 Exploiting $>_{2/3}$-majorities

As shown in the previous subsection, for $s < 3/4$ we cannot provide a linear problem kernel with respect to the "number of dirty pairs" by adapting the 3/4-Majority Rule. The following three lemmas establish the basis for an alternative polynomial-time data reduction rule to obtain fixed-parameter tractability with respect to the "number of dirty pairs". Note that the following results will not improve the fixed-parameter tractability results with respect to the "average KT-distance". However, the "number of dirty pairs" according to the $\geq_s$-majority for values of $s < 3/4$ provides a "stronger" parameterization than the "average KT-distance" in the sense that it might allow for smaller parameter values for the same instance. To this end, recall that the number of dirty pairs according to the $\geq_{3/4}$-majority can be much higher than the the number of dirty pairs according to the $>_{2/3}$-majority.

We state the result for the $>_{2/3}$-majority. Clearly, it holds for any $\geq_s$-majority with $s > 2/3$. The basic idea is to consider an order that is induced by the $>_{2/3}$-majorities of the nondirty pairs and then to show that a dirty candidate can only "influence" the positions of nondirty candidates that are not "too far away" from it in this order. Then, it is safe to remove nondirty candidates that cannot be influenced by any dirty candidate. In the following, let D denote the set of dirty candidates and n_d denote the number of dirty pairs according to the $>_{2/3}$-majority in an election.

Lemma 4.3. *For an election containing n_d dirty pairs, in every Kemeny consensus at most n_d nondirty pairs are not ordered according to their $>_{2/3}$-majorities.*

Proof. For an election (V, C) with n_d dirty pairs, let l be a preference list with $P := \{\{c, c'\} \mid c > c' \text{ in } l \text{ and } c' >_{2/3} c\}$ and $|P| > n_d$. We show that l cannot be optimal.

Let $l_{2/3}$ denote a preference list with $c > c'$ for all pairs with $c >_{2/3} c'$ and the remaining dirty pairs are ordered arbitrarily. First, we show that such an order exists. Due to Proposition 4.1, all nondirty candidates can be ordered according to the $>_{2/3}$-majority order. Analogously, one can show that every dirty candidate can be ordered according to the 2/3-majority with respect to all nondirty candidates and that two dirty candidates that form a nondirty pair do not violate transitivity if ordered according to the 2/3-majority of this pair. Since the remaining dirty pairs can be ordered arbitrarily, they can be ordered without violating transitivity as well.

We show that $\text{score}(l) > \text{score}(l_{2/3})$. Let C_P denote the set of all candidate pairs of C, that is, $C_P := \{\{c, c'\} : c, c' \in C,\ c \neq c'\}$, and D_P denote the set of all dirty pairs in (V, C). Then, $\text{score}(l)$ and $\text{score}(l_{2/3})$ can be decomposed into partial scores depending on candidate pairs of P, D_P, and $C_P \backslash (D_P \cup P)$, that is,

$$\text{score}(l) = s_l(P) + s_l(D_P) + s_l(C_P \backslash (D_P \cup P)).$$

Now, consider score(l) − score$(l_{2/3})$. Since all pairs $p \in C_P \backslash (D_P \cup P)$ are ordered according to the $>_{2/3}$-majority in l and in $l_{2/3}$, the partial scores for them are equal. The partial score for every nondirty pair is more than $2n/3$ if it is not ordered according to the $>_{2/3}$-majority, and less than $n/3$ otherwise. Together with the fact that for a dirty pair the difference of the partial scores of the two possible orders is at most $n/3$, one obtains

$$s_l(D_P) - s_{l_{2/3}}(D_P) \geq -|D_P| \cdot n/3,$$

and

$$s_l(P) - s_{l_{2/3}}(P) > |P| \cdot n/3.$$

Since $|P| > |D_P|$, it follows that score(l) − score$(l_{2/3}) > n/3 > 0$. Thus, l cannot be optimal. □

In the following, we show that the bound on the number of "incorrectly" ordered nondirty pairs from Lemma 4.3 can be used to fix the relative order of two candidates forming a nondirty pair. For this, it will be useful to have a concept of distance of candidates with respect to the order induced by the $>_{2/3}$-majority. For an election (V, C) and a nondirty pair $\{c, c'\}$, define

$$\text{dist}(c, c') := \begin{cases} |\{b \in C : b \text{ is nondirty and } c >_{2/3} b >_{2/3} c'\}| & \text{if } c >_{2/3} c' \\ |\{b \in C : b \text{ is nondirty and } c' >_{2/3} b >_{2/3} c\}| & \text{if } c' >_{2/3} c. \end{cases}$$

Lemma 4.4. *Let (V, C) be an election and let $\{c, c'\}$ be a nondirty pair. If* dist$(c, c') \geq n_d$*, then in every Kemeny consensus $c > c'$ iff $c >_{2/3} c'$.*

Proof. Let l be a preference list such that there is a nondirty pair $\{c, c'\}$ with $c > c'$ in l, $c' >_{2/3} c$, and dist$(c, c') \geq n_d$. We show that l cannot be a Kemeny consensus. Since dist$(c, c') \geq n_d$, there is a set E of at least n_d nondirty candidates with $c' >_{2/3} e >_{2/3} c$ for $e \in E$. Since $c > c'$ in l, the candidates from E cannot be ordered according to the $>_{2/3}$-majority with respect to c or c' in l. Hence, there are at least n_d pairs formed by the candidates from E and c or c' in l, which, together with the pair $\{c, c'\}$, give more than n_d nondirty pairs that are not ordered according to the $>_{2/3}$-majority. This contradicts Lemma 4.3 and, thus, l cannot be optimal. □

Finally, the next lemma enables us to fix the position in a Kemeny consensus for a nondirty candidate that has a sufficiently large distance to all dirty candidates.

Lemma 4.5. *If for a nondirty candidate c it holds that* dist$(c, c_d) > 2n_d$ *for all dirty candidates $c_d \in D$, then c is ordered according to the $>_{2/3}$-majority with respect to all candidates from C in every Kemeny consensus.*

Proof. Assume that there is a nondirty candidate c with dist$(c, c_d) > 2n_d$ for all $c_d \in D$ and that there is a preference list l with $e > c$ for a candidate e with $c >_{2/3} e$. Then, we show that l cannot be optimal.

Since dist$(c, c_d) > 2n_d$ for all dirty candidates $c_d \in D$, it follows from Lemma 4.4 that all dirty candidates must be ordered according to the $>_{2/3}$-majority with respect to c. Thus, e must be a nondirty candidate. Due to Lemma 4.4, dist$(e, c) < n_d$. Since for all $c_d \in D$ one has dist$(c, c_d) > 2n_d$, it follows from dist$(e, c) < n_d$ that dist$(e, c_d) > n_d$ for all $c_d \in D$ as well. Thus, in a Kemeny consensus, e must

be ordered according to the $>_{2/3}$-majority with respect to all dirty candidates due to Lemma 4.4. For a candidate $c_d \in D$ one has $c >_{2/3} c_d$ iff $e >_{2/3} c_d$ since for all $c_d \in D$ one has $\text{dist}(c, c_d) > 2n_d$ and $\text{dist}(e, c) < n_d$. Hence, there is no dirty candidate $c_d \in D$ with $e > c_d > c$ in l, that is, all candidates $f_i, i = 1, \ldots, s$, with $e > f_i > \cdots > f_s > c$ in l must be nondirty. Then, analogously to the proof of Proposition 4.1, one can show that ordering $c, e, f_1, \ldots, f_s$ according to the $>_{2/3}$-majority gives a consensus with score less than the score of l. Thus, l cannot be optimal. □

The correctness of the following data reduction rule follows directly from Lemma 4.5. It is not hard to verify that it can be carried out in $O(n \cdot m^2)$ time.

Rule 4.2. For an election with n_d dirty pairs, let c be a nondirty candidate with $\text{dist}(c, c_d) > 2n_d$ for all $c_d \in D$. Let $C_l := \{c' \in C : c' >_{2/3} c\}$ and $C_r := \{c' \in C : c >_{2/3} c'\}$. Delete c and reorder every vote such that $C_l > C_r$ and the order of the candidates within C_l and C_r remains unchanged.

In the following, we show that after exhaustively applying Rule 4.2, the number of nondirty candidates is bounded by a function quadratic in the "number of dirty pairs".

Theorem 4.3. *For $s > 2/3$,* Kemeny score *admits a partial kernel with at most $2n_d + 8n_d^2$ candidates where n_d denotes the number of dirty pairs.*

Proof. An instance with n_d dirty pairs has at most $2n_d$ dirty candidates. For every nondirty candidate c not deleted after exhaustively applying Rule 4.2, there must be a dirty candidate c_d with $\text{dist}(c, c_d) \leq 2n_d$. Thus, for every dirty candidate there can be at most $4n_d$ nondirty candidates that are not deleted. It follows that, in total, there can be at most $2n_d \cdot 4n_d$ nondirty candidates left. The theorem follows. □

4.4 Conclusion

We conclude this chapter with an overview of the provided results, a short a discussion of the applicability of the introduced framework to Kemeny Score with Ties, and finally state some open questions deriving directly from our results.

Overview of the results. Our results are summarized in Table 4.2. We identified a concept of dirtiness leading to some observations of structural properties of a Kemeny consensus (see Table 4.1, Section 4.2). These observations provided the basis for the identification of a polynomial-time solvable special case and data reduction rules resulting in partial kernelization results. The new concept of partial kernelization may significantly ease the task to develop provably effective data reduction rules for multidimensional problems where it seems difficult to provide "full" kernel results. For Kemeny Score, this would include the development of data reduction rules that provably decrease the number of votes.

Finally, note that due to the dependencies of d_a and $\overline{d}$ as discussed in Section 3.8 our results directly transfer to $\overline{d}$.

Table 4.2: Partial kernelization results for Kemeny Score. The term dirty refers to the $\geq_s$-majority according to the respective values of s. The number of dirty pairs is n_d^s and d_a denotes the average KT-distance. An instance is nondirty if it does not contain any dirty pair.

value of s	results
$1/2 \leq s \leq 2/3$	-
$2/3 < s < 3/4$	polynomial-time solvability for nondirty instances (Proposition 4.1) quadratic partial kernel wrt. n_d^s (Theorem 4.3)
$3/4 \leq s \leq 1$	polynomial-time solvability for nondirty instances (Proposition 4.1) linear partial kernel wrt. d_a and wrt. n_d^s (Theorem 4.1)

Kemeny Score with Ties. First parameterized complexity results for Kemeny Score with Ties with respect to several parameterizations have been discussed in Section 3.7. The question of fixed-parameter tractability of Kemeny Score with Ties with respect to the "average KT-distance" has been left open. This question can be answered positively since the new method for partial kernelization introduced in Section 4.1 also applies to Kemeny Score with Ties [25]. To this end, we extend the definition of dirtiness as follows. A pair of candidates a, b is dirty if neither $a >_s b$ nor $a =_s b$ nor $a <_s b$ according to a $\geq_s$-majority where one has $a =_s b$ if $a = b$ in at least sn votes. Using analogous but more laborious proofs as in this chapter, one can show the following results [25].

- A Kemeny Score with Ties instance without dirty pairs is solvable in polynomial time.
- Kemeny Score with Ties admits a quadratic partial kernel with respect to the "average KT-distance" as well with respect to the "number of dirty pairs".

Open Problems. The results presented in this chapter lead to several concrete questions.

- Despite the negative results from Theorem 4.2, there is still room for improving the $>_{2/3}$-majority based results. In particular, is there a linear partial kernel with respect to the $\geq_s$-majority for any $s < 3/4$? A natural step in answering this question seems to investigate whether for two *nondirty* candidates a, b, there must be a Kemeny consensus with $a > b$ if $a \geq_s b$.
- A challenging task of theoretical interest concerns the development of classical problem kernels also bounding the number of votes for Kemeny Score with and without ties.
- We introduced the new structural parameters "number of dirty candidates" and "number of dirty pairs". The investigation of further fixed-parameter algorithms with respect to these parameterizations is clearly of interest. This is especially motivated by the observation that there are instances in which the "dirtiness" parameters assume small values whereas the parameters "number of candidates",

"average/maximum KT-distance" and "average distance from the Kemeny consensus" can be arbitrarily large. For example, consider the election consisting of the vote

$$a_1 > a_2 > \cdots > a_m$$

and three identical votes defined as follows

$$a_m > a_{m-1} > \cdots > a_1.$$

There is no dirty pair according to the $\geq_{3/4}$-majority but the values of the other three parameters grow at least linearly in m.

- For Kemeny Score with Ties there are only studies according to the $>_{2/3}$-majority resulting in a quadratic partial kernel with respect to the average KT-distance [25]. It seems very promising that a linear partial kernel can be obtained analogously to the case without ties by using the $\geq_{3/4}$-majority.

Finally, we stress that partial kernelization might be of interest for many NP-hard problems defined on elections. For example, Conitzer [57] uses a different notion of similarity to efficiently compute the closely related Slater rankings. Using a concept of similar candidates, he identifies efficiently solvable special cases, yielding a powerful preprocessing technique for computing Slater rankings. It is interesting to investigate if the concept of (partial) kernelization might be used to provide some performance guarantee of the corresponding reduction rules.

The following chapter provides experimental results showing the usefulness of data reduction for the computation of a Kemeny consensus.

Chapter 5

Experimental results for Kemeny

We investigated the practical value of fixed-parameter algorithms for computing optimal Kemeny rankings. Our main focus was on data reduction rules leading to partial kernelization as described in Chapter 4. To this end, we implemented and extended the 3/4-Majority Rule introduced in Subsection 4.3.1. In addition, we implemented the search tree algorithm from Section 3.4, the dynamic programming algorithm showing fixed-parameter tractability with respect to the number of candidates (Section 3.3) as well as an ILP-based algorithm used in previous experimental work [59, 185]. We showed that the data reduction rules allow for the computation of Kemeny rankings of instances that cannot be solved by the other implemented algorithms without data reduction.

Combining our data reduction with the other implemented algorithms, we provide encouraging results in experiments with real-world data arising in web search and sport competitions. We often achieve provably optimal rankings with small running times—for example, a few seconds or even milliseconds for instances with about 100–150 candidates. An essential property of our data reduction algorithm is that it can break instances into several subinstances to be handled independently, that is, the relative order between the candidates in two different subinstances in a Kemeny ranking is already determined. This also means that for many of the instances which could not be completely solved, we were still able to compute "partial rankings" of the top and bottom ranked candidates. For example, for a large instance based on rankings of about 1300 mathematicians according to their impact in the world wide web, we could not compute a complete Kemeny ranking but still provide a "partial" ranking of the best 31 mathematicians.

In our experiments, we are not only interested in the decision problem Kemeny Score but also want to compute a corresponding Kemeny ranking. Hence, we deal with the following NP-hard optimization problem.

Rank aggregation
Input: An election (V, C).
Task: Find a Kemeny ranking of (V, C).

Our algorithms for Rank aggregation are implemented in C++ and the source

INPUT: An election (V, C).
OUTPUT: A minimal subset $C' \subseteq C$ with $c' \geq_{1/2} c$ for every $c' \in C'$ and every $c \in C \setminus C'$.

For every candidate $c \in C$
  Start with $M_c := \{c\}$.
    Repeat until M_c remains unchanged
      If there is a candidate $c' \in M_c$ and a candidate $c'' \in C \setminus M_c$ with $c'' >_{1/2} c'$,
      then add c'' to M_c.

Return: an M_c such that $|M_c| \leq |M_{c'}|$ for every $c' \in C \setminus \{c\}$.

Figure 5.1: Strategy to find winning subsets.

code and test data are available under the GPL Version 3 license[1]. In the following two sections, we first provide more details on the implemented algorithms and then describe our experimental results.

5.1 Implemented algorithms

In this section, we describe the algorithms realized in our software package. We distinguish between data reduction rules and other "solution algorithms".

5.1.1 Data reduction rules

We present a well-known data reduction rule of practical relevance and show that it reduces an instance at least as much as the 3/4-Majority Rule (see Subsection 4.3.1). The reduction rule is based on the following easy-to-verify observation.

Observation 5.1. *Let $C' \subseteq C$ be a candidate subset with $c' \geq_{1/2} c$ for every $c' \in C'$ and every $c \in C \setminus C'$. Then there must be a Kemeny ranking fulfilling $C' > C \setminus C'$.*

To turn Observation 1 into a reduction rule, we need a polynomial-time algorithm to identify appropriate "winning subsets" of candidates. We use the following simple strategy, called *winning subset routine* provided in Figure 5.1.

Condorcet-Set Rule. *If the winning subset routine returns a subset C' with $C' \neq C$, then replace the original instance by the two subinstances induced by C' and $C \setminus C'$.*

It is easy to see that the Condorcet-Set Rule can be carried out in $O(nm^3)$ time. The following proposition shows that the Condorcet-Set Rule is at least as powerful as the 3/4-Majority Rule, implying that the Condorcet-Set Rule provides a partial kernel with less than $11d_a$ candidates.

Proposition 5.1. *An instance reduced by the Condorcet-Set Rule cannot be further reduced by the 3/4-Majority Rule.*

Proof. The proof is by contradiction. Assume that there is an instance reduced by the Condorcet-Set Rule where the 3/4-Majority Rule successfully applies. Then, there must be a non-dirty candidate n_i and a subset $C' \subseteq C$ with $c' \geq_{3/4} n_i$ for $c \in C'$ and

[1] Download from `http://theinf1.informatik.uni-jena.de/kconsens/`

$n_i \geq_{3/4} c$ for $c \in C \setminus (C' \cup \{n_i\})$. Clearly, we can assume that C' and $C \setminus (C' \cup \{n_i\})$ are nonempty since otherwise the Condorcet Set Rule would obviously reduce this instance. Hence, in the iteration loop of the routine given in Figure 5.1 for n_i all candidates from C' must be added to M_{n_i}. Since $|\{v \in V : c' \geq_{3/4} n_i\}| \geq 3/4 \cdot |V|$ and $|\{v \in V : n_i \geq_{3/4} c\}| \geq 3/4 \cdot |V|$, the intersection of these two sets must contain at least $|V|/2$ elements, that is, there must be at least $|V|/2$ votes with "$c' > \cdots > n_i > \cdots > c$" for $c \in C'$ and $c \in C \setminus (C' \cup \{n_i\})$. Hence, M_{n_i} is a minimal subset according to which the instance can be split according to the Condorcet-Set Rule, a contradiction to the fact that it has been applied exhaustively. □

Proposition 5.1 shows that the 3/4-Majority Rule cannot lead to a "stronger" reduction of an instance than the Condorcet-Set Rule does. However, since the Condorcet-Set Rule has a higher running time, that is $O(nm^3)$ compared to $O(nm^2)$, applying the 3/4-Majority Rule before the Condorcet-Set Rule may lead to an improved running time in practice. Analogously, this is also true for the following "special case" of the Condorcet-Set Rule also running in $O(nm^2)$ time.

Condorcet Rule. *If there is a candidate $c \in C$ with $c \geq_{1/2} c'$ for every $c' \in C \setminus \{c\}$, then delete c.*

Indeed, our experiments will show that combining the Condorcet-Set Rule with the other rules significantly speeds up the practical running times for many instances.

Further data reduction rules. In addition to the two Condorcet rules and the 3/4-Majority Rule, we implemented and evaluated three further reduction rules. The $>_{2/3}$-majority based Rule 4.2 from Subsection 4.3.3, a reduction rule which replaces a set of candidates by a weighted candidate behaving in exactly the same way to all other candidates, and, a reduction rule generalizing the 3/4-Majority Rule by using "non-dirty sets" of candidates. Since none of these rules in our experiments led to a stronger reduction of the instance size or to an improved running time for any instance, we omit them from further considerations.

5.1.2 Exact solution algorithms

To optimally solve subinstances remaining after data reduction and to investigate how far one can get without data reduction, we implemented an integer linear programming-based algorithm and two fixed-parameter algorithms.

Fixed-parameter algorithms. We decided to implement two of the "simpler" fixed-parameter algorithms from Chapter 3.

- We implemented the search tree algorithm branching into conflict tuples (Section 3.4) for tuple sizes ranging from two to six. For this algorithm, the worst-case running time bound depending on the Kemeny score is very rough for many instances and hence there is reasonable hope that the algorithm might perform better in practice. Moreover, the search tree algorithm only needs polynomial space whereas all dynamic programming algorithms provided in Chapter 3 require exponential space.

- We implemented the dynamic programming algorithm exploiting the parameter "number of candidates" m running in $O(2^m \cdot nm^2)$ time (Section 3.3). Since for all our instances m is smaller than d_a, this seems to be a good choice based on comparing the worst-case running times.

The implementation of some of the other fixed-parameter algorithms such as the search tree strategies provided by Simjour [189] is an important task of future research.

Integer linear programming. Regarding previous approaches to compute optimal Kemeny rankings, the state of the art seems to be to formulate an integer linear program (ILP) and solve it with a standard solver. Conitzer et al. [59] provided different ILP formulations and gave practical evidence (based on random data and using CPLEX) that the best of them outperforms a previous branch and bound algorithm from [68]. Hence, we use the corresponding formulation [59, Linear Program 3] to compare it with our algorithms. Herein, we use the freely available ILP-solver GLPK[2]. The same formulation was also used to compute optimal solutions to experimentally investigate the performance of several heuristics and approximation algorithms [185].

5.2 Experimental results

Our algorithms are implemented in C++ using several libraries of the boost package. Our implementation consists of about 4000 lines of code. All experiments were carried out on a PC with 3 GHz and 4 GB RAM (CPU: Intel Core2Quad Q9550) running under Ubuntu 9.10 (64 bit) Linux.

We start to describe our results for instances obtained from sport competitions followed by two different types of web search data.[3] In general, for the smaller instances (most of the sport instances and some of the web instances) we focus on comparing the ILP-based algorithm with the dynamic programming and the search tree variants for different tuple sizes. For the larger instances (which cannot be solved without reduction rules) we give a systematic analysis of the performance of the individual reduction rules.

5.2.1 Sport competitions

Formula 1. The winner determination of a Formula 1 season can be considered as an election where the candidates are the drivers and the votes are the single races. Under the current system the winner determination is based on a "scoring rule", that is, in a single race every candidate gets some points depending on the outcome and the candidate with highest total score wins. We computed Kemeny winners for the seasons from 1970 till 2008. Since currently our implementation cannot handle ties, we only considered candidates that have competed in all races. Candidates that dropped out of a race are ordered according to the order determined by how long the drivers participated in the race. The generated instances have about 16 votes and up to 28 candidates.

[2] http://www.gnu.org/software/glpk/

[3] Our dataset can be found under `http://theinf1.informatik.uni-jena.de/kconsens/`.

Without data reduction, the ILP-approach was the most successful algorithm. It could solve all instances in less than 31 seconds whereas the dynamic programming algorithm could not solve the two instances with the highest number of candidates within five minutes. All search tree variants performed even worse. Our reduction rules partitioned nearly all instances in very small components such that a Kemeny ranking could be computed for all years except 1983 in few milliseconds. For 1983 (24 candidates), a remaining component with 19 candidates could be solved in less than one minute by the dynamic programming algorithm.

The Kemeny winner in most of the considered seasons is the same as the candidate selected by the used scoring rule. However, in 2008, Lewis Hamilton was elected as world champion (beating Felipe Massa by only one point) whereas Massa was the "Condorcet driver" and thus the candidate ranked first in every Kemeny ranking.

Winter sport competitions. For ski jumping and cross skiing, we considered the world cup rankings from the seasons 2005/2006 to 2008/2009,[4] ignoring candidates not appearing in all four rankings. Without data reduction, the ski jumping instance, consisting of 33 candidates, was solved by GLPK in 103 seconds whereas the search tree and dynamic programming algorithms did not find a solution within five minutes. Only using the Condorcet Rules, the instance was solved in milliseconds. The cross skiing instance, consisting of 69 candidates, could not be solved without data reduction within five minutes by any of the solution algorithms but was reduced by our reduction rules in 0.04 seconds such that one component with 12 and one component with 15 candidates were left and all other positions could be determined by the reduction rules. The remaining components could be solved for example by the dynamic programming algorithm within 0.12 and 0.011 seconds.

5.2.2 Search result rankings

A prominent application of Rank Aggregation is the aggregation of search result rankings obtained from different web search engines. We queried the same 37 search terms as Dwork et al. [77] and Schalekamp and van Zuylen [185] to generate rankings. We used the search engines Google, Lycos, MSN Live Search, and Yahoo! to generate rankings of 1000 candidates. Note that most of the search engines do not return more than 1000 search terms. We consider two search results as identical if their URL is identical up to some canonical form (cutting after the top-level domain). Results not appearing in all rankings are ignored. We end up with 36 instances having between 55 and 163 candidates and omit the instance corresponding to the search term "zen budism" with only 18 candidates from further considerations. We start with a systematic investigation of the performance of the individual reduction rules followed by describing our results for the web instances.

A useful notation to display the effects of the reduction rules are *profiles* explained at the following example. For the search term "architecture" the profile describing the reduced instance is

$$1^{36} > 12 > 1^{30} > 17 > 1^{27}.$$

Every "1" stands for a position for which a candidate was determined in a Kemeny ranking and higher numbers for groups of candidates whose "internal" order could

[4] Obtained from `http://www.sportschau.de/sp/wintersport/`

Table 5.1: The first column encodes the combination of reduction rules used: the first digit is "1" if the Condorcet-Set Rule is applied, the second if the Condorcet Rule is applied and the last digit is "1" if the 3/4-Majority Rule is applied. For the three instances corresponding to the search terms blues, gardening, and classical guitar we give the running times in seconds and profiles describing the result of the data reduction process.

	blues		**gardening**	
	time	profile	time	profile
001	0.03	$1^2 > 5 > 1 > 101 > 1 > 2$	0.01	$1 > 2 > 1 > 102$
010	0.10	$1^{74} > 9 > 1^{29}$	0.05	$1^{54} > 43 > 1^9$
011	0.10	$1^{74} > 9 > 1^{29}$	0.05	$1^{54} > 43 > 1^9$
100	0.84	$1^{74} > 9 > 1^{29}$	0.95	$1^{54} > 20 > 1^3 > 9 > 1^{10} > 4 > 1^6$
101	0.10	$1^{74} > 9 > 1^{29}$	1.03	$1^{54} > 20 > 1^3 > 9 > 1^{10} > 4 > 1^6$
110	0.10	$1^{74} > 9 > 1^{29}$	0.10	$1^{54} > 20 > 1^3 > 9 > 1^{10} > 4 > 1^6$
111	0.10	$1^{74} > 9 > 1^{29}$	0.11	$1^{54} > 20 > 1^3 > 9 > 1^{10} > 4 > 1^6$

	classical guitar	
	time	profile
001	0.03	$1 > 114$
010	0.06	$1^6 > 92 > 1^{17}$
011	0.07	$1^6 > 92 > 1^{17}$
100	1.89	$1^6 > 7 > 1^{50} > 35 > 1^{17}$
101	2.03	$1^6 > 7 > 1^{50} > 35 > 1^{17}$
110	0.19	$1^6 > 7 > 1^{50} > 35 > 1^{17}$
111	0.18	$1^6 > 7 > 1^{50} > 35 > 1^{17}$

not be determined by the data reduction rules. Sequences of i ones are abbreviated by 1^i. That is, we know the order of the best 36 candidates, then we know the set of candidates that must assume positions 37– 48 without knowledge of their relative orders, and so on.

We systematically applied all combinations of reduction rules, always sticking to the following rule ordering: If applied, the Condorcet-Set Rule is applied last and the 3/4-Majority Rule is applied first. After a successful application of the Condorcet-Set Rule, we "jump" back to the other rules (if "activated"). Examples are given in Table 5.1. This led to the following observations.

- Surprisingly, the Condorcet Rule alone led to a stronger reduction than the 3/4-Majority Rule in most of the instances whereas the 3/4-Majority Rule never led to a stronger reduction than the Condorcet Rule.

- For several instances the Condorcet Set Rule led to a stronger reduction than the other two rules, for example, for gardening and classical guitar (see Table 5.1). More specifically, it led to a stronger reduction for 14 out of the 36 instances. Furthermore, restricted to the 15 instances with more than 100 candidates (given

Table 5.2: Web data instances with more than 100 candidates. The first column denotes the search term, the second the number of candidates, the third the running time in seconds, and the last column the profiles remaining after data reduction.

search term	# cand.	time	structure of reduced instance		
affirmative action	127	0.21	1^{27}	$> 41 >$	1^{59}
alcoholism	115	0.10	1^{115}		
architecture	122	0.16	1^{36}	$> 12 > 1^{30} > 17 >$	1^{27}
blues	112	0.10	1^{74}	$> 9 >$	1^{29}
cheese	142	0.20	1^{94}	$> 6 >$	1^{42}
classical guitar	115	0.19	1^{6}	$> 7 > 1^{50} > 35 >$	1^{17}
Death+Valley	110	0.11	1^{15}	$> 7 > 1^{30} > 8 >$	1^{50}
field hockey	102	0.17	1^{37}	$> 26 > 1^{20} > 4 >$	1^{15}
gardening	106	0.10	1^{54}	$> 20 > 1 > 1 > 9 > 1^{8} > 4 >$	1^{9}
HIV	115	0.13	1^{62}	$> 5 > 1^{7} > 20 >$	1^{21}
lyme disease	153	3.08	1^{25}	$> 97 >$	1^{31}
mutual funds	128	2.08	1^{9}	$> 45 > 1^{9} > 5 > 1 > 49 >$	1^{10}
rock climbing	102	0.07	1^{102}		
Shakespeare	163	0.26	1^{100}	$> 10 > 1^{25} > 6 >$	1^{22}
telecommuting	131	1.60	1^{9}	$> 109 >$	1^{13}

in Table 5.2), it led to a stronger reduction for eight of them.

- The running times for the Condorcet-Set Rule in combination with the other rules are given in left part of Figure 5.2. Applying the Condorcet Rule before the Condorcet-Set Rule led to a significant speed-up. Additionally applying the 3/4-Majority Rule changes the running time only marginally. Note that jumping back to the "faster" rules after applying the Condorcet-Set Rule is crucial to obtain the given running times.

In the following, by "our reduction rules", we refer to all three rules applied in the order: Condorcet Rule, 3/4-Majority Rule, and Condorcet-Set Rule.

Now, we describe our overall results. For all instances with more than 100 candidates, the results of our reduction rules are displayed in Table 5.2: the data reduction rules are not only able to reduce candidates at the top and the last positions but also partition some instances into several smaller subinstances. Out of the 36 instances, 22 were solved directly by the reduction rules and one of the other algorithms in less than five minutes. Herein, the reduction rules always contributed with less than four seconds to the running time. For all other instances we still could compute the "top" and the "flop" candidates of an optimal ranking. For example, for telecommuting there remains a subinstance with 109 candidates but we know the best nine candidates (and their order). The effectiveness in terms of top candidates of our reduction rules combined with the dynamic programming algorithm is illustrated in Figure 5.2. For example, we were able to compute the top seven candidates for all instances and the top 40 candidates for 70 percent of the instances.

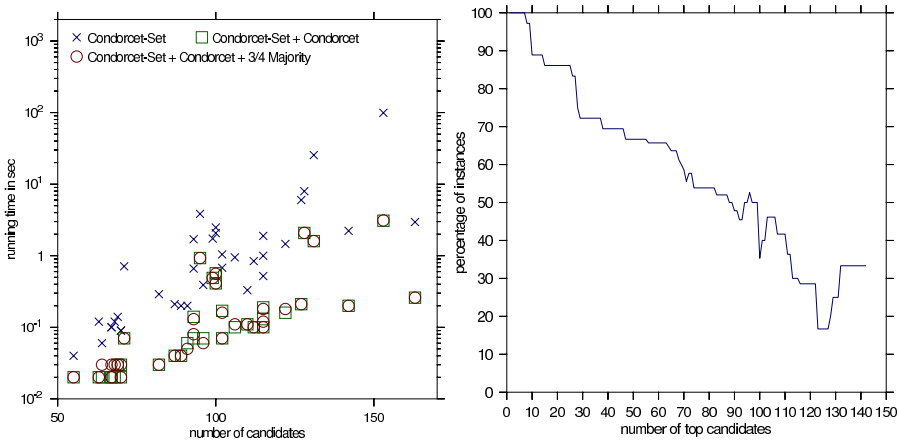


Figure 5.2: Left: Running times of different combinations of reduction rules. Right: Percentage of the web data instances for which the x top candidates could be determined by data reduction and dynamic programming within five minutes. For a given number x of top positions, we only considered instances with at least x candidates.

5.2.3 Impact rankings

Since most search engines only display the best 1000 search results, we could not generate instances with a higher number of candidates using the approach from the previous subsection. To generate instances with more candidates, we generated rankings that measure the "impact in the web" of different search terms. For a search engine, a list of search terms is ranked according to the number of the hits of each single term. We used Ask, Google, MSN Live Search, and Yahoo! to generate rankings for all capitals, all nations, and the 103 richest people of the world.[5] Our biggest instance is built from a list of 1349 mathematicians.[6]

As to the capitals, in less than a second, our algorithms (reduction rules and any of the other algorithms for solving subinstances up to 11 candidates) computed the following "profile" of a Kemeny ranking: $1^{45} > 34 > 1^{90} > 43 > 1^{26}$. The final Kemeny ranking starts as follows: London > Paris > Madrid > Singapore > Berlin > $\cdots$. For aggregating the nation rankings, our algorithms were less successful. However, we could still compute the top 6 and the flop 12 candidates. Surprisingly, the best represented nation in the web seems to be Indonesia, followed by France, the United States, Canada, and Australia. The instance consisting of the 103 richest persons could be solved exactly in milliseconds by the Condorcet Rules. In contrast, for the mathematicians we could only compute the top 31 and flop 31 candidates but could not deal with a subinstance of 1287 candidates between. For the mathematicians instance, the search strategy for minimal subsets for the Condorcet-Set Rule (Figure 5.1) led to a running time of more than a day. Hence, we used a cutoff of 20 candidates for the size of the minimal subsets. This decreased the running time to less than one hour. Searching for sets with up to 100 candidates did not improve the results but increased

[5] http://en.wikipedia.org/wiki/List_of{capitals_by_countries, richest_people}
[6] http://aleph0.clarku.edu/~djoyce/mathhist/chronology.html

the running time to about one day. Note that with this cutoff Proposition 5.1 no longer holds, that is, the 3/4-Majority Rule is provably no longer subsumed by the modified Condorcet-Set Rule. However, for the mathematicians instance even an extended version of the 3/4-Majority Rule did not lead to an improved result.

5.3 Conclusion

Our experiments showed that the corresponding data reduction rules allow for the computation of optimal Kemeny rankings for real-world instances of non-trivial sizes within seconds. In contrast, all of our larger instance (with more than 50 candidates) could not be solved by ILP-formulation, the previously fastest exact algorithm [59], or the two other implemented fixed-parameter algorithms directly. A crucial observation in the experiments with the different reduction rules regards some cascading effects, that is, jumping back to the faster rules after a successful application of the Condorcet-Set Rule does significantly improve the running time showing that the order of reduction rules is relevant. We mention that we could not observe a specific behavior for the different types of data. However, a further extension of the data set and experiments in this direction are clearly of interest. We end with some remarks leading to tasks for future research.

Comparison of the three solution algorithms. Although all three solution algorithms were clearly outperformed by the data reduction rules in the sense that without data reduction only relatively small instances could be tackled, they still turned out useful to efficiently solve remaining parts of the instances. We briefly compare their performances. The ILP seems to be the most effective in general but it might be worth trying the search tree algorithm in parallel since they are significantly faster than the ILP algorithm for some instances. Moreover, practical experiments [59] showed that the running time for the ILP increases if the instances become more "disturbed", also implying a higher average KT-distance. In contrast, the performance of the dynamic programming algorithm seems to be unaffected by the average KT-distance. However, since the data reduction rules usually cut away "easy parts", it seems reasonable that the remaining subinstances will come with a higher average KT-distance than the original instance. Thus, the dynamic programming algorithm (maybe with some additional heuristic speed-up) might perform better for them. Since for our test data the number of remaining unsolved components of reasonable sizes (e.g. between 25 and 45 candidates) is too small (less than 10) to allow for a systematic study, such investigations are deferred to future work.

Finding all Kemeny rankings. For some applications it is desirable to obtain *all* Kemeny rankings. Due to Lemma 4.2, the 3/4-Majority Rule preserves all Kemeny rankings and hence there is also a linear partial kernel with respect to the "average KT-distance" d_a for the enumeration variant of Kemeny Score.[7] In contrast to the 3/4-Majority Rule, the Condorcet rules do not preserve all optimal solutions. However, both Condorcet rules can be adapted easily to this case by replacing "$c \geq_{1/2} c'$" by "$c >_{1/2} c'$" in their statements. Some preliminary experiments showed that this made

[7] Simjour [189] provided a search-based enumeration algorithm running in $32^{d_a} \cdot \text{poly}(n, m)$ time.

the rules significantly less effective for the web instances consisting of only four votes. In contrast, for instances consisting of more than four votes such as the Formula 1 instances, the reduction rules did not perform as well as before but still led to a significant reduction. Hence, the development of further reduction rules preserving all Kemeny rankings seems to be an interesting task for future research.

Parameter-based evaluation of heuristics. Schalekamp and van Zuylen [192] investigated the performance of a wide range of approximation algorithms for two different data sets. It might be interesting to extend this study by investigating how the performance of different heuristics or approximation algorithms relates to certain instance properties. For example, which heuristic performs best for instances with large average KT-distance and which performs best for small average KT-distance?

Constraint rankings. An important extension of Rank Aggregation is to consider "constraint rankings", that is, the problem input additionally contains a prespecified order of some candidate pairs in the consensus list [192]. Here, our data reduction rules cannot be applied anymore. Developing new reduction rules for this scenario is of great interest since they also could be used in combination with the search tree algorithm in an "interleaving mode" [171, 173]. Herein, the basic idea is to apply the reduction rules after every branching step.

Improved partial kernel. The Condorcet-Set Rule is at least as effective as the 3/4-Majority Rule (Proposition 5.1). Can this be used to provably shrink the partial kernel size? Also note that the development of further data reduction rules based on a $\geq_s$-majority for $s < 3/4$ might lead to reduction rules not subsumed by the Condorcet-Set Rule.

Concluding, our experiments provided first encouraging results for computing optimal Kemeny rankings by using fixed-parameter algorithms. The implementation of some of the other fixed-parameter algorithms and extended algorithm engineering efforts are important tasks for future research. For a given election, the values of most of the different parameters investigated for Kemeny Score (Table 3.2, Section 3.8) can either be computed optimally or approximated by a constant factor in polynomial time. This invites for the development of a "meta-algorithm" computing the different parameter values of an instance and then deciding about the appropriate fixed-parameter algorithm to apply to this input instance. Since our theoretical results are based on worst-case analysis, the development of such an algorithm clearly needs thorough experimental validations based on different data sets. However, this is a promising approach in the spirit of multivariate algorithmics [104, 172].

Chapter 6

Dodgson and Young voting

The well-known Condorcet principle from 1785 [69] requires that a winner of an election is the candidate who is preferred to each other candidate in more than half of the votes. Unfortunately, such a Condorcet winner does not always exist. Hence, several voting systems have been proposed which always choose the Condorcet winner if one exists, and, otherwise, pick a candidate that is in some sense closest to being a Condorcet winner. In other words, these election systems deal with certain "editing problems". We focus on two classic editing problems from social choice theory, one due to C. L. Dodgson[1] from 1876 [71] and one due to H. P. Young from 1977 [203]. In Dodgson elections, the editing operation is to switch neighboring candidates in the voters' preference lists and the *Dodgson score* of a candidate is the minimum number of switches needed to make this candidate a Condorcet winner. In Young elections, the editing operation is to remove a vote. For any candidate, its *dual Young score* denotes the minimum number of removals needed to make it a Condorcet winner and its *Young score* denotes the number of remaining votes.

In a seminal work, Bartholdi et al. [12] initiated the study of the computational complexity of the winner determination for some election systems. They showed that to decide whether a distinguished candidate can be made a Condorcet winner by performing no more than a given number of editing operations is NP-complete for both Dodgson and Young elections. In a further breakthrough, for Dodgson elections Hemaspaandra et al. [130] and later for Young elections Rothe et al. [183] showed that the corresponding winner and ranking problems are even complete for $\mathbf{P}_{\parallel}^{\mathbf{NP}}$, the class of problems that can be solved via parallel access to NP. On the algorithmic side, there is a simple greedy heuristic for finding Dodgson winners with a guaranteed frequency of success [134] and some work on the polynomial-time approximability of Dodgson and Young elections [50, 164]. In particular, Caragiannis et al. [50] gave (randomized) approximation algorithms for Dodgson elections and showed that it is hard to approximate Young elections by any factor. In further work, Caragiannis et al. [51] investigated the development of "socially desirable approximations" for Dodgson's rule aiming to fix some of the shortcomings of Dodgson's rule as a choice procedure [40]. Summarizing, Dodgson's rule is one of the most studied voting rules

[1] Also known as the writer Lewis Carroll.

Table 6.1: Parameterized complexity of Dodgson Score and (Dual) Young Score with respect to different parameters. In case of fixed-parameter tractability we also give information about the (exponential terms of the) corresponding running times. Bold-faced results are provided in this chapter, the FPT results for the parameter "number of candidates" can be directly obtained from [12, 203], and the FPT results for Young elections with respect to the number of votes are trivial. For Young elections, the same approaches can be used for Dual Young Score and Young Score for n and m, respectively. W[1]-hardness with respect to the number of votes for Dodgson Score was proven by Fellows et al. [106].

Parameter	Dodgson Score	Dual Young Score	Young Score
# votes n	W[1]-hard	FPT (2^n)	
# candidates m	FPT (ILP+Lenstra)	FPT (ILP+Lenstra)	
# steps k	**FPT (2^k)**	**W[2]-complete**	**W[2]-complete**

in computational social choice. In the following, we compare it with Young's rule concerning the parameterized complexity of the corresponding decision problems.

Table 7.1 provides an overview of the parameterized complexity for the problem of deciding about the score of a distinguished candidate for Dodgson and Young elections. For the standard parameter "number of candidates" the fixed-parameter tractability results follow from ILP-formulations [12, 203] and Lenstra's results (see Subsection 1.3.3). For Dodgson Score this fixed-parameter tractability result has been further refined by McCabe-Dansted [163]. The W[1]-hardness for Dodgson Score with respect to the number of votes has been recently obtained by Fellows et al. [106]. For a constant number of votes, Dodgson Score is solvable in polynomial time [12].

This chapter investigates the the parameterized complexity with respect to the parameter "number of editing operations" for which, other than in the classical context, the parameterized complexity of Dodgson and Young elections differs (see Table 7.1). Dodgson Score is fixed-parameter tractable with respect to the "number of switches". In contrast, deciding whether a distinguished candidate can become a winner by deleting a certain number of votes is W[2]-complete with respect to the "number of deleted votes" (Dual Young Score) as well as with respect to the "number of remaining votes" (Young Score). Our results imply that Dodgson elections can be put into actual use whenever the input instances are close to having a Condorcet winner. This answers an open question of Christian et al. [56][2] and refutes a parameterized hardness conjecture of McCabe-Dansted [163]. In addition to our results, there is an exponential-size problem kernel for a generalized variant of Dodgson Score [108] which is complemented by a recent result of Fellows et al. [106] showing that under some reasonable assumption from classical complexity theory Dodgson Score does not admit a problem kernel of polynomial-size[3].

We now define the basic computational problems of this chapter. Examples are provided in Figure 6.1. A *switch* denotes the swapping of two neighboring candidates

[2]Fellows and Rosamond independently showed that Dodgson Score is fixed-parameter tractable, but with a higher running time.

[3]We refer to Subsection 9.2.2 for more details about the underlying framework.

Input	Dodgson	Young
$v_1 : c > a > b$	$c > a > b$	$c > a > b$
$v_2 : c > a > b$	$c > a > b$	$c > a > b$
$v_3 : a > b > c$	$a > \mathbf{c} > \mathbf{b}$	$\cancel{a > b > c}$
$v_4 : b > c > a$	$b > c > a$	$\cancel{b > c > a}$
$v_5 : b > c > a$	$b > c > a$	$b > c > a$

Figure 6.1: Example for Dodgson and Young score. For the election provided by $\{v_1, \ldots, v_5\}$ (left-hand side), the candidate c can become a Condorcet winner by applying one switch in v_3 (middle) or by deleting two votes (right-hand side). Thus, the Dodgson score of c is one, the Young score of c is three, and the dual Young score is two.

in a vote.

> Dodgson Score:
> *Given*: An election (V, C), a distinguished candidate $c \in C$, and an integer $k \geq 0$.
> *Question*: Can c be made a Condorcet winner by at most k switches?

In other words, for Dodgson Score, we ask whether the *Dodgson score* of c is at most k. The *Young score* is defined by the number of remaining votes:

> Young Score:
> *Given*: An election (V, C), a distinguished candidate $c \in C$, and an integer $l \geq 0$.
> *Question*: Is there a subset $V' \subseteq V$ of size at least l such that (V', C) has the Condorcet winner c?

The *dual Young score* is defined by the number of removed votes:

> Dual Young Score:[4]
> *Given*: An election (V, C), a distinguished candidate $c \in C$, and an integer $k \geq 0$.
> *Question*: Is there a subset $V' \subseteq V$ of size at most k such that $(V \backslash V', C)$ has the Condorcet winner c?

All three problems are NP-complete [12, 183]. We briefly discuss that, other than in Figure 6.1, for many instances the Dodgson score of a candidate is expected to be higher than the (dual) Young Score since the switch operation is less powerful than deleting votes in the sense that a switch affects only two candidates whereas deleting a vote affects all candidates. For example, consider the election formed by the two votes

$$c > a_1 > a_2 > \cdots > a_s \text{ and } a_1 > a_2 > \cdots > a_s > c.$$

Then, the Dodgson score of c is s since c must be switched to the first position in the second vote. In contrast, the Young score and the dual Young score are one

[4] The Dual Young Score problem can also be considered as constructive control by deleting votes in order to make a distinguished candidate the Condorcet winner. We refer to Chapter 11 for more details about control in elections.

vote 1 :	c_2	$>$	c_3	$>$	c	$>$	c_1			
vote 2 :	c_1	$>$	c_2	$>$	c	$>$	c_3	Deficits:	candidate c_1:	$d_1 = 1$
vote 3 :	c_1	$>$	c_2	$>$	c	$>$	c_3		candidate c_2:	$d_2 = 3$
vote 4 :	c_2	$>$	c	$>$	c_3	$>$	c_1		candidate c_3:	$d_3 = 0$
vote 5 :	c_3	$>$	c_2	$>$	c_1	$>$	c			

Figure 6.2: An election and the corresponding deficits of the candidates c_1, c_2, and c_3 against the distinguished candidate c. For example, c_2 is preferred to c in each of the five votes and hence c must "improve" upon c_2 in at least three of the votes to become a Condorcet winner.

since it is sufficient to delete the second vote. Hence, the "parameterized tractability gap" between Dodgson and Young elections with respect to the "number of editing operations" is not completely surprising.

In the following section, we discuss our results for Dodgson Score including considerations of a generalized model. More specifically, in case of allowing sets of tied candidates in a vote, depending on the choice between two switching mechanisms, we either obtain fixed-parameter tractability or W[2]-completeness.

6.1 Dodgson Score

In this section, we describe a fixed-parameter algorithm based on dynamic programming for the problem Dodgson Score parameterized by the score k with running time $O(2^k \cdot nk + nm)$. This answers an open question of Christian et al. [56]. The dynamic programming algorithm will be stated for the decision problem but can easily be extended such that for a yes instance it stores a sequence of at most k switches leading to an election in which the distinguished candidate is a Condorcet winner. In the following, we first describe a general version of the algorithm, which we also use to solve a generalized version of Dodgson Score (see Subsection 6.1.2). Then we show how to further improve the running time of this algorithm for Dodgson Score.

6.1.1 Dynamic programming algorithm

To design our algorithm, we make use of the following easy-to-verify observation [163, Lemma 2.19].

Observation 1. There is always an optimal solution consisting of a sequence of switches such that every switch moves the distinguished candidate c to a better position.

Making use of Observation 1, our algorithm only considers switches of this kind. To "measure" how much the distinguished candidate c must improve upon any candidate $c' \in C \setminus \{c\}$, we introduce the concept of the *deficit* of c': Let $N_{c'}$ denote the number of votes from V in which c' *defeats* c, that is, in which c' is better positioned than c. Then, the *deficit* $d_{c'}$ is $\lfloor (N_{c'} - (n - N_{c'}))/2 \rfloor + 1$, that is, the minimum number of votes in which the relative order of c and c' has to be reversed such that c defeats c' in strictly more than half of the votes. See Figure 6.2 for an example. We call a candidate with a positive deficit *dirty*. Now, we can state a further observation which

	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$	$(0,2)$	$(1,2)$	$(0,3)$	$(1,3)$
$\{v_1\}$	∞	∞	∞	∞	∞	2	∞	0
$\{v_1, v_2\}$	...	...						
$\{v_1, v_2, v_3\}$	...							

Figure 6.3: Dynamic programming table T for the election given in Figure 6.2 with dirty candidates c_1 and c_2 and corresponding deficit list $(d_1, d_2) = (1, 3)$. The first row can be initialized as follows. Without any switch the deficit list is clearly $(1, 3)$ and applying two switches in v_1 decrements the deficit of c_2 by one providing the entry for $(1, 2)$. All other deficit lists can not be achieved by only switching within v_1.

will be crucial for the analysis of the algorithm when bounding the size of the dynamic programming table.

Observation 2. A candidate with nonpositive deficit can never become dirty through a sequence of switches since we consider only switches that never increase any deficit (Observation 1). One switch decreases the deficit of one candidate by one. Therefore, with at most k switches allowed, in a yes-instance, the sum of the deficits of the dirty candidates is bounded from above by k.

Observation 2 allows us to restrict our attention to the set of dirty candidates. Let $C_d = (c_1, c_2, \ldots, c_p)$ denote the list of dirty candidates in an arbitrary but fixed order and let $D = (d_1, d_2, \ldots, d_p)$ be the corresponding *deficit list*. We define a two-dimensional dynamic programming table T, each row corresponding to a subset of votes $\{v_1, v_2, \ldots, v_i\}$ for $i = 1, \ldots, n$ and each column corresponding to a *(partial) deficit list* $(d'_1, d'_2, \ldots, d'_p)$ with $0 \leq d'_j \leq d_j$ for $1 \leq j \leq p$ (see Figure 6.3 for an example). The entry

$$T(i, (d'_1, d'_2, \ldots, d'_p))$$

stores the minimum number of switches within $\{v_j \mid 1 \leq j \leq i\}$ such that in a resulting instance the deficits of the p dirty candidates are at most $d'_1, d'_2, \ldots, d'_p$, respectively.[5] If a deficit list $(d'_1, d'_2, \ldots, d'_p)$ cannot be achieved by switching within the set of votes $\{v_j \mid 0 \leq j \leq i\}$, then $T(i, (d'_1, d'_2, \ldots, d'_p)) := +\infty$. Then, $T(n, (0, 0, \ldots, 0)) \leq k$ implies that within the set of all n votes a sequence of at most k switches can be applied such that the distinguished candidate becomes a Condorcet winner and hence the considered instance is a yes-instance.

Using that the sum of the deficits of the dirty candidates is bounded from above by the parameter k (Observation 2), we will show later that for a yes-instance, one has to consider at most 2^k different deficit lists and hence that size of T is bounded by a function depending only on the parameter. In the following, we describe how to "fill" the table T. We give some further definitions which are necessary to describe the initialization and the update step.

Let switch(v_i, c_j) denote the minimum number of switches needed such that in vote v_i candidate c defeats candidate c_j. If c already defeats c_j in v_i, then switch$(v_i, c_j) :=$ 0. For a deficit list $D' = (d'_1, d'_2, \ldots, d'_p)$ and a subset of indices $S \subseteq \{1, \ldots, p\}$, we use

[5] Using "at most" in the definition of table entries, we do not have to consider deficit lists $(d'_1, \ldots, d'_p)$ where $d'_i < 0$ for some i. In this way, the case that an optimal solution may decrease the deficit of a dirty candidate to a negative value is also covered.

Algorithm *DodScore*
Input: Set of votes $V = \{v_1, \ldots, v_n\}$, set of candidates C, set of dirty candidates $C_d = \{c_1, \ldots, c_p\} \subseteq C$, distinguished candidate c, deficit list $D = (d_1, \ldots, d_p)$ of dirty candidates, positive integer k with $\sum_{i=1}^{p} d_i \leq k$
Output: Yes, if c can become a Condorcet winner with at most k switches

Initialization:
01 for all $D' = (d'_1, \ldots, d'_p)$ with $0 \leq d'_j \leq d_j$ for $0 \leq j \leq p$
02 for $i = 1, \ldots, n$
03 $T(i, D') := +\infty$
04 for all $S \subseteq \{1, \ldots, p\}$
05 if for each $j \in S$ candidate c_j defeats c in v_1 then
06 $T(1, D - S) := \text{switch}(v_1, \text{best}(S, v_1))$
Update:
07 for $i = 2, \ldots, n$
08 for all $D' = (d'_1, \ldots, d'_p)$ with $0 \leq d'_j \leq d_j$ for $0 \leq j \leq p$
09 for all $S \subseteq \{1, \ldots, p\}$
10 if for each $j \in S$ candidate c_j defeats c in v_i then
11 $T(i, D') := \min\{T(i, D'),\ T(i-1, D' + S) + \text{switch}(v_i, \text{best}(S, v_i))\}$

Output:
12 if $T(n, (0, 0, \ldots, 0)) \leq k$ then
13 return "Yes"

Figure 6.4: Fixed-parameter algorithm with respect to the "number of switches" for Dodgson Score.

$D' + S$ to denote a deficit list $(e_1, \ldots, e_p)$ where $e_i := d'_i + 1$ for $i \in S$ and $d'_i < d_i$, and $e_i := d'_i$, otherwise. Analogously, for the original deficit list $D = (d_1, \ldots, d_p)$, $D - S$ denotes the list $(f_1, \ldots, f_p)$ where $f_i := d_i - 1$ if $i \in S$ and $f_i := d_i$, otherwise. Let $\text{best}(S, v_i)$ denote the candidate c_j with $j \in S$ such that c_j is liked better than each other candidate in $\{c_r \mid r \in S, r \neq j\}$ in vote v_i.

The dynamic programming algorithm for Dodgson Score is stated in Figure 6.4. We assume that the deficits of the candidates are provided as input and that the sum of the deficits of the dirty candidates is at most k as argued in Observation 2. In the initialization of the first row of the dynamic programming table (Figure 6.4, lines 4–6), the algorithm considers all possible combinations of *deficit decrements* that can be achieved by switches within the first vote, and stores the minimum number of switches needed for each of them. An example is provided in Figure 6.3. In the update (lines 7–11), the subset of votes $\{v_1, \ldots, v_{i-1}\}$ is extended by a new vote v_i and for the new subset $\{v_1, \ldots, v_i\}$ a solution for all partial deficit lists is computed by combining a number of switches within the new vote v_i with information already stored in the table T. See Figure 6.5 for an example of the update step.

Lemma 6.1. *The algorithm DodScore (Figure 6.4) is correct.*

Proof. Concerning the correctness of the initialization, note that the first for-loop (lines 1–3) merely sets all table entries to "$+\infty$". Hence it suffices to show that *DodScore* assigns the correct number of switches to all entries of the first row with

	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$	$(0,2)$	$(1,2)$	$(0,3)$	$(1,3)$
$\{v_1\}$	∞	∞	∞	∞	∞	2	∞	0
$\{v_1, v_2\}$	∞	4	∞	3	2	1	2	0
$\{v_1, v_2, v_3\}$	...							

$$T(2,(1,2)) = \min \begin{cases} T(1,(1,2)), \\ T(1,(1,3)) + \text{cost of improvement in } v_2 \end{cases}$$

Figure 6.5: Example for the update step for the election provided in Figure 6.2. Consider the highlighted table entry which is computed according to the formula given below the table. Since the cost of the improvement from $(1,3)$ to $(1,2)$ in v_2 is one and $T(1,(1,3)) = 0$, the updated value of the considered entry is one.

partial deficit lists that can be achieved by switching within the first vote v_1 (lines 4–6). Since in one vote the deficit of every candidate can be reduced by at most one, it is sufficient to iterate over all possible subsets S of $\{1,\ldots,p\}$ and to reduce the original deficits of the corresponding candidates by one. Thereby, an entry can only become less than $+\infty$ if c can be improved upon all candidates with indices in S (line 5). Moreover, the minimum number of switches is obviously the number of switches needed to improve c upon the candidate that is best in vote v_1 among the candidates c_j with $j \in S$, which is given by $\text{switch}(v_1, \text{best}(S, v_1))$.

The computation of an entry $T(i, D')$ with $i \geq 2$ is based on the fact that the decrement from D to D' can be split into two parts. One part needs to be achieved by switches in vote v_i and the other one by switches in votes $v_1,\ldots,v_{i-1}$. The minimum number of switches needed for the corresponding splitting possibilities is stored in $T(i, D')$. Moreover, since the switches in v_i can decrease the deficit of one dirty candidate by at most one, every possible way of splitting the deficit decrement can be represented by a subset S of $\{1,\ldots,p\}$. Each subset S has the meaning that, by the switches in v_i, the deficits of the dirty candidates with indices in S should be decreased by exactly one; the rest of the decrement from D to D' has to be achieved by switches in $v_1,\ldots,v_{i-1}$. According to the definition of the table T, the minimum number of switches to achieve the latter is stored in the already computed $(i-1)$th row of T, namely, in $T(i-1, D'+S)$. As argued for the initialization, $\text{switch}(v_i, \text{best}(S, v_i))$ returns the minimum number of switches to decrease the deficit of the candidates with indices in S. Therefore, lines 9-11 of *DodScore* compute $T(i, D')$ correctly.

Since *DodScore* computes the table T correctly, we can conclude that a given instance is a yes-instance if and only if $T(n,(0,\ldots,0)) \leq k$ (lines 12 and 13). □

Lemma 6.2. *The algorithm DodScore (Figure 6.4) runs in $O(4^k \cdot nk + nm)$ time.*

Proof. It is easy to see that the deficit list D can be computed in $O(nm)$ time by iterating over all votes and counting the deficits for all candidates. Now, we consider the size of the dynamic programming table.

A deficit d'_i can have values ranging from 0 to d_i. Hence, the number of partial deficit lists, that is, the number of columns in the table, is $\prod_{i=1}^{p}(d_i+1)$. Clearly, for a potential yes-instance, we have the constraints $p \leq k$ and $\sum_{i=1}^{p} d_i \leq k$ (see

Observations 1 and 2). It is not hard to see that 2^k is a tight upper bound on $\prod_{i=1}^{p}(d_i+1)$. Thus, the overall table size is $n \cdot 2^k$.

For computing the value of a table entry $T(i, D')$, the algorithm iterates over all 2^p subsets of $\{1, \ldots, p\}$. For each such subset S, it computes the "distance" in v_i between the best of the dirty candidates with indices in S and c, that is, the number of switches needed to make c better than this best dirty candidate. This distance can be computed in $O(k)$ time and, hence, the computation of $T(i, D')$ can be done in $O(2^k \cdot k)$ time. The initialization of T clearly needs $O(2^k \cdot n)$ time. Hence, table T can be computed in $O(2^k \cdot n \cdot 2^k \cdot k + 2^k \cdot n) = O(4^k \cdot nk)$ time. □

By making use of a "monotonicity property" of the table, we can improve the running time of *DodScore* as shown in the following theorem.

Theorem 6.1. Dodgson Score *can be solved in* $O(2^k \cdot nk + nm)$ *time.*

Proof. The improvement compared to Lemma 6.2 is achieved by replacing the innermost for-loop (lines 9–11 in Figure 6.4) of the update step which computes a table entry and needs $O(2^k \cdot k)$ time by an instruction running in time linear in k.

For $d \in C \backslash \{c\}$, let $S_i(d)$ denote the set of the dirty candidates that are better than the distinguished candidate c but not better than the candidate d in vote v_i. Clearly, $S_i(d)$ is empty if d is worse than c in v_i and, otherwise, $S_i(d)$ contains d. We replace lines 9–11 in Figure 6.4 by the recurrence

$$T(i, D') := \min_{1 \le r \le p} \{T(i-1, D' + S_i(c_r)) + \text{switch}(v_i, c_r)\}.$$

To prove the correctness of the recurrence, on the one hand, observe that, for every r with $1 \le r \le p$, there exists a subset $S \subseteq \{1, \ldots, p\}$ satisfying the if-condition in line 10 of *DodScore* such that $S = S_i(c_r)$ and $\text{best}(S, v_i) = c_r$. Thus,

$$\begin{aligned} & \min_{S \subseteq \{1,\ldots,p\}} \{T(i-1, D' + S) + \text{switch}(v_i, \text{best}(S, v_i))\} \\ \le \quad & \min_{1 \le r \le p} \{T(i-1, D' + S_i(c_r)) + \text{switch}(v_i, c_r)\}. \end{aligned}$$

On the other hand, for every $S \subseteq \{1, \ldots, p\}$ satisfying the if-condition in line 10, there exists an r with $1 \le r \le p$ such that $S \subseteq S_i(c_r)$. For instance, let r be the index of the candidate in S that is the best in v_i; we then have $\text{best}(S, v_i) = c_r$ and, thus, $\text{switch}(v_i, \text{best}(S, v_i)) = \text{switch}(v_i, c_r)$. Moreover, from the definition of table entries, the following monotonicity of the table T is easy to verify:

$$T(i, (d_1, \ldots, d_i, \ldots, d_p)) \ge T(i, (d_1, \ldots, d_i + 1, \ldots, d_p)).$$

Thus, from $S \subseteq S_i(c_r)$ we conclude that $T(i, D' + S) \ge T(i, D' + S_i(c_r))$. Clearly, $S_i(c_r) \subseteq \{1, \ldots, p\}$ and, by definition, $S_i(c_r)$ satisfies the if-condition in line 10. It follows that

$$\begin{aligned} & \min_{1 \le r \le p} \{T(i-1, D' + S_i(c_r)) + \text{switch}(v_i, c_r)\} \\ \le \quad & \min_{S \subseteq \{1,\ldots,p\}} \{T(i-1, D' + S) + \text{switch}(v_i, \text{best}(S, v_i))\}. \end{aligned}$$

The time for computing a table entry in the improved version is clearly $O(k)$: Before looking for the minimum, we can compute $S_i(c_r)$ for all $1 \le r \le p$ by iterating one time over v_i. Then, based on Lemma 6.2, the overall running time becomes $O(2^k \cdot nk + nm)$. □

6.1.2 Allowing ties

Sometimes it might be desirable to allow a voter to rank two or more candidates equally. This leads to an election based on votes with ties. As noted by Hemaspaandra et al. [130], there are (at least) two different natural models on how to generalize Dodgson Score to the case with ties. The models differ in the "power" of one switch. In the first model, transforming $a = b > c$ into $c > a = b$ requires just one switch and in the second model this requires two separate switches. The ranking and the winner versions remain $\mathrm{P}_{||}^{\mathrm{NP}}$-complete in both cases [130].

Formally, a vote with ties can be considered as a total order of disjoint sets of candidates. To ease the presentation, we often write just "$> c >$" instead of "$> \{c\} >$" in case of a singleton.

Recall that in the case of ties a candidate c is a Condorcet winner if for every other candidate d the number of votes in which c is strictly preferred to d is higher than the number of votes in which d is strictly preferred to c. Hence, the deficit of a candidate $d \neq c$ is defined as $N_d - N_{\overline{d}} + 1$, where N_d is the number of votes in which d defeats c and $N_{\overline{d}}$ is the number of votes in which c defeats d. In the following, we describe two switch operations, one for each model. In both models a switch can now either break or build ties between the distinguished candidate c and other candidates.

For computing the Dodgson score only the relative order between the distinguished candidate c and the other candidates is relevant. Hence, to keep the models easy, we restrict them to the interesting case where each switch involves the distinguished candidate.

In the first model the distinguished candidate can improve upon a whole subset of candidates by one switch. More precisely, for an appropriate subset $B \subseteq C \backslash \{c\}$, we have one of the following two situations:

- "$\ldots > B > c > \ldots$": Such a vote can be transformed to "$\ldots > B \cup \{c\} > \ldots$" by applying one switch.
- "$\ldots > B \cup \{c\} > \ldots$": Such a vote can be transformed to "$\ldots > c > B > \ldots$" by applying one switch.

The problem of computing the Dodgson score for this model is denoted as Dodgson Tie Score 1 (DTS1).

In the second model, the switch operation becomes less powerful, that is, the distinguished candidate can only improve upon one candidate by one switch. Here, one has to consider the following situations:

- "$\ldots > B > c > \ldots$": Such a vote can be transformed to "$\ldots > B \backslash B' > B' \cup \{c\} > \ldots$" by $|B'|$ switches for any $B' \subseteq B$.
- "$\ldots > B \cup \{c\} > \ldots$": Such a vote can be transformed to "$\ldots > (B \backslash B') \cup \{c\} > B' \ldots$" by $|B'|$ switches for any $B' \subseteq B$.

The problem of computing the Dodgson score for this model is denoted as Dodgson Tie Score 2 (DTS2). The considered model is very general in the sense that it allows to choose to improve the distinguished candidate only upon a subset of equally ranked candidates and thus it is only "charged" to pay for this subset. A reasonable special case of this model is to restrict B' to be identical with B, that is, to allow only

to switch the distinguished candidate with the whole subset. For this case, we can directly use the improved version of the algorithm *DodScore* as described in the proof of Theorem 6.1 by treating the whole set of tied candidates as one possibility. This yields an algorithm with running time $O(2^k \cdot nk + nm)$.

Note that for both models, a switch as defined for the case without ties can be simulated by two switches.

Whereas DTS1 and DTS2 remain NP-complete (which easily follows from the NP-completeness of the case without ties [12]), their parameterized complexity differs. The problem DTS2 is fixed-parameter tractable while DTS1 is W[2]-complete. We briefly discuss the corresponding results and refer to the journal paper [27] for the full proofs.

The fixed-parameter tractability of DTS2 can be obtained by a slight modification of algorithm *DodScore* from Figure 6.4. Since we do not have a total ordering of candidates in the votes, we cannot make use of the monotonicity property employed in the proof of Theorem 6.1. Thus, returning to the algorithm used for Lemma 6.4, for DTS2 we obtain a slightly worse running time than for Dodgson Score without ties as given in Theorem 6.1. More specifically, this leads to the following.

Theorem 6.2. Dodgson Tie Score 2 *can be solved in* $O(6^k \cdot nk + nm)$ *time.*

In contrast to the fixed-parameter tractability of DTS2, DTS1 is W[2]-complete with respect to k [27, Theorem 3]. Intuitively, this may be explained by the fact that in case of DTS1 a single edit operation can improve the distinguished candidate c upon, in principle, *all* other candidates.

6.2 Young Score

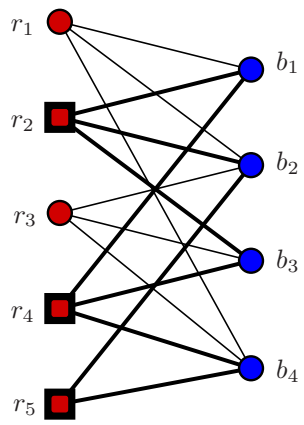
In this section, we show that Young Score and Dual Young Score are W[2]-complete with respect to their corresponding solution size bounds l and k, respectively. From a parameterized perspective Dual Young Score appears to be more natural than Young Score because for Dual Young Score one may expect smaller parameter values.

For both problems, similar to Dodgson Score, it is helpful to consider a deficit concept for a candidate $d \in C \backslash \{c\}$ against the distinguished candidate c: Again, let N_d denote the number of votes from V in which d defeats c. Then, the *Young deficit* is defined as $N_d - (n - N_d)$.

We start with a W[2]-hardness-proof for Dual Young Score, giving two parameterized reductions: The first reduction is from the W[2]-hard Red Blue Dominating Set (RBDS) [76] to an intermediate problem, which is a variant of Red Blue Dominating Set, and then the second one is from the intermediate problem to Dual Young Score.

> Red Blue Dominating Set (RBDS)
> *Given:* A bipartite graph $G = (R \cup B, E)$, with R and B being the two disjoint vertex sets, and an integer $k \geq 0$.
> *Question:* Is there a subset $D \subseteq R$ of size at most k such that every vertex in B has at least one neighbor in D?

The intermediate problem is defined as follows (see Figure 6.6 for an example).



b_1 is dominated by r_2 and r_4
b_2 is dominated by r_2 and r_5
b_3 is dominated by r_2 and r_4
b_4 is dominated by r_4 and r_5

Figure 6.6: Instance of $k/2$-Red Blue Dominating Set with red vertices $r_1, r_2, \ldots, r_5$ and blue vertices $b_1, b_2, \ldots, b_4$. For the case $k = 3$, that means one needs to find three red vertices such that every blue vertex is dominated at least $\lfloor 3/2 \rfloor + 1 = 2$ times, a solution is provided by $\{r_2, r_4, r_5\}$.

$k/2$-Red Blue Dominating Set ($k/2$-RBDS)
Given: A bipartite graph $G = (R \cup B, E)$, with R and B being the two disjoint vertex sets, and an integer $k \geq 0$.
Question: Is there a subset $D \subseteq R$ of size at most k such that every vertex in B has at least $\lfloor k/2 \rfloor + 1$ neighbors in D?

Lemma 6.3. *$k/2$-Red Blue Dominating Set is W[2]-hard.*

Proof. We give a parameterized reduction from RBDS. Let $(G = (B \cup R, E), k)$ denote an RBDS instance. A corresponding instance $(G' = (B' \cup R', E'), k')$ of $k/2$-RBDS is constructed as follows:

$$\begin{aligned}
B' &:= B \cup \{b_x\}, \\
R' &:= R \cup \{r_j^{\text{new}} \mid 1 \leq j \leq k\} \cup \{r_x\}, \\
E' &:= E \cup \{\{b, r_j^{\text{new}}\} \mid b \in B \text{ and } 1 \leq j \leq k\} \\
&\quad \cup \{\{b_x, r_x\}\} \cup \{\{b_x, r_j^{\text{new}}\} \mid 1 \leq j \leq k\}, \text{ and} \\
k' &:= 2k + 1.
\end{aligned}$$

The following claim finishes the proof.

Claim: The considered RBDS-instance is a yes-instance if and only if the $k/2$-RBDS-instance is a yes-instance.

"$\Rightarrow$": One can easily construct a solution for the $k/2$-RBDS-instance by choosing the corresponding vertices of the size-$\leq k$ RBDS-solution D and additionally the $k+1$ new red vertices. The size of the new solution then is at most $2k+1$ and every blue vertex in B is dominated k times by the new red vertices and at least once by a vertex from D. The new blue vertex b_x is dominated by the $k+1$ new red vertices. Therefore, every vertex is dominated at least $k + 1 = \lfloor k'/2 \rfloor + 1$ times.

"$\Leftarrow$": Consider a size-$\leq k'$ solution D of the $k/2$-RBDS-instance. Obviously, D must contain r_x and the other k new red vertices $r_1^{\text{new}}, \ldots, r_k^{\text{new}}$ to dominate b_x. Therefore, all of the other blue vertices are dominated exactly k times by $r_1^{\text{new}}, \ldots, r_k^{\text{new}}$. Since every blue vertex has to be dominated at least $\lfloor k'/2 \rfloor + 1 = k + 1$ times, the vertices in $D \setminus \{r_x, r_1^{\text{new}}, \ldots, r_k^{\text{new}}\}$ dominate all other blue vertices in $B' \setminus \{b_x\} = B$ and, thus, the subset of R corresponding to $D \setminus \{r_x, r_1^{\text{new}}, \ldots, r_k^{\text{new}}\}$ is a size-$\leq k$ solution of the RBDS-instance. □

Next, we give a parameterized reduction from $k/2$-RBDS to Dual Young Score.

Lemma 6.4. Dual Young Score *is W[2]-hard.*

Proof. Given a $k/2$-RBDS-instance $(G = (B \cup R, E), k)$ with $B = \{b_1, ..., b_m\}$ and $R = \{r_1, ..., r_n\}$,[6] we first consider the case that k is odd. The corresponding Dual Young Score instance is constructed as follows. We set $C := \{c_i \mid b_i \in B\} \cup \{a, b, c\}$. Let

$$N_C(r_i) := \{c_j \in C \mid \{r_i, b_j\} \in E\}$$

and

$$\overline{N_C(r_i)} := C \setminus (\{a, b, c\} \cup N_C(r_i)),$$

that is, the candidates in $N_C(r_i)$ correspond to the neighbors of r_i in G and $\overline{N_C(r_i)}$ corresponds to the rest of the vertices in B. Construct three disjoint subsets of votes, V_1, V_2, and V_3:

- The votes in V_1 correspond to the red vertices in R. For every red vertex r_i, add a vote v_i to V_1 in which the candidates in $N_C(r_i) \cup \{a, b\}$ are better than c and the candidates in $\overline{N_C(r_i)}$ are worse than c. More precisely,

  $$V_1 := \{b > a > N_C(r_i) > c > \overline{N_C(r_i)} \mid 1 \leq i \leq n\}.$$

  Recall that, if there is a set of candidates in a vote, then the order of the elements in the set is assumed to be fixed arbitrarily.

- The set V_2 also contains n votes. These votes guarantee that in $V_1 \cup V_2$ the deficit of b is $2k - 2$ whereas the deficit of each other candidate is zero.

  $$\begin{aligned} V_2 \quad := \quad & \{\overline{N_C(r_i)} > c > N_C(r_i) > b > a \mid 1 \leq i \leq n - k + 1\} \\ & \cup \{b > \overline{N_C(r_i)} > c > N_C(r_i) > a \mid n - k + 2 \leq i \leq n\}. \end{aligned}$$

  As we will see later, the $(2k - 2)$-deficit of b is needed to ensure that all votes in a solution of a Dual Young Score instance have to belong to V_1.

- The set V_3 consists of $k - 1$ votes adjusting the deficits of a and b so that in $V_1 \cup V_2 \cup V_3$ both a and b have a deficit of $k - 1$ and all other candidates have a deficit of 0. Let $C_R := C \setminus \{a, b, c\}$. The set V_3 consists of $\lfloor k/2 \rfloor$ votes with $a > C_R > c > b$ and $\lfloor k/2 \rfloor$ votes with $a > c > C_R > b$.

Finally, the overall set V of votes is $V_1 \cup V_2 \cup V_3$ and the upper bound for the solution size of the Dual Young Score instance is set to k. The key idea behind

[6] Within this proof, n does not denote the number of votes.

Figure 6.7: Parameterized reduction from $k/2$-RBDS to Dual Young Score. In this example, $n = 5$ and $k = 3$, that is, every blue vertex b_i must be dominated by at least two red vertices from $\{r_1, \ldots, r_5\}$. For the bipartite graph given on the left-had side this can be achieved by the red vertices r_2, r_4, and r_5. Deleting the corresponding votes v_{r2}, v_{r4}, and v_{r5} in the election sketched on the right-hand side implies that for every "blue candidate" b_i, at least two votes in which b_i is preferred to c are removed. Based on an appropriate placement of the blue candidates within the votes from $V_2 \cup V_3$, this ensures that c is a winner in the resulting election.

the above construction is that to reduce the $(k-1)$-deficits of a and b by deleting at most k votes, every solution of the Dual Young Score instance consists of *exactly* k votes from V_1. The reason for this is that the votes in V_1 are the only votes whose deletion simultaneously reduces the deficits of a and b against c (see Figure 6.7).

In the following, we show that c can become the Condorcet winner by deleting at most k votes if and only if there is a dominating set of size at most k for the (G, k).

"$\Rightarrow$": Every solution V' of Dual Young Score must contain exactly k votes from V_1 and, by the above construction, each vote in V_1 corresponds to a vertex in R. Denote the corresponding subset of R by D. Since V' is a solution, every candidate $c_i \in (C \setminus \{a, b, c\})$ must be better than c in at least $\lfloor k/2 \rfloor + 1$ of the votes in V'. Therefore, choosing the corresponding red vertices to form a dominating set implies that every blue vertex is dominated at least $\lfloor k/2 \rfloor + 1$ times.

"$\Leftarrow$": Since every dominating set $D \subseteq R$ of size at most k dominates each blue vertex at least $\lfloor k/2 \rfloor + 1$ times, we can easily extend D to a dominating set D' of size exactly k by adding $k - |D|$ arbitrary red vertices to D. Since every red vertex corresponds to a vote from V_1, we thus obtain a size-k subset V' of V corresponding to D'. According to the above construction of V_1, the removal of V' results in a new vote set where the deficits of a and b are both -1 and the deficits of all other candidates are ≤ -1. Therefore, c can become the Condorcet winner by deleting exactly k votes.

Recall that in the definition of V_3 it is decisive that k is odd. Now, we consider the case that k is even and give a reduction from Dual Young Score with an odd k to Dual Young Score with an even k. Given a Dual Young Score instance (V, C, c, k) with k being odd, we add a new vote v to V that has the form: "$C \setminus \{c\} > c$" to get the new vote set V'. Then $(V', C, c, k' := k + 1)$ is a Dual Young Score instance with k' being even. The correspondence between the solutions is easy to achieve. $\square$

To prove that Dual Young Score is W[2]-complete, it remains to show its containment in W[2]. This can be done by devising a reduction to the Optimal Lobbying problem which has been shown to be W[2]-complete by Christian et al. [56]. We refer to [27, Lemma 7] for the corresponding proof showing containment in W[2] for Dual Young Score. Altogether, one arrives at the following.

Theorem 6.3. Dual Young Score *is W[2]-complete.*

Using a similar reduction as the one in the proof of [27, Lemma 7] (containment in W[2]) and a parameterized version of the nonparameterized reduction from the W[2]-hard Set Packing problem to Young Score as presented by Rothe et al. [183, Theorem 2.3] (W[2]-hardness), we can also derive the following theorem.

Theorem 6.4. Young Score *is W[2]-complete.*

6.3 Conclusion

The most important observation to derive from the results of this chapter is that Dodgson and Young elections behave differently with respect to the parameter "number of editing operations". Whereas for Dodgson elections we achieve fixed-parameter tractability, we experience parameterized intractability in case of Young elections. This stands in sharp contrast to traditional complexity analysis, where both election systems appear as equally hard [12, 130, 183], and complements results on polynomial-time approximability [50]. Furthermore, we found that the complexities of Dodgson elections allowing ties between the candidates strongly vary (fixed-parameter tractability vs W[2]-completeness) depending on the cost model for switching ties. Again, in the standard complexity framework these two cases cannot be differentiated because both lead to NP-completeness.

We conclude with some remarks and open questions for future research.

- In Subsection 6.1.2, we introduced two different models on how to deal with ties for Dodgson Score. This setting can be further generalized to allow incomplete votes (partial orders) as input. In contrast to Kemeny's system which can be easily adapted to this case (see Section 3.7), for Dodgson's system it is not clear how to proceed in this scenario. In particular, it is not clear how to define a meaningful switch operation on partial orders. Hence, the development of an appropriate model seems to be of interest.

  Young elections can be directly extended to incomplete votes since the edit operation "deleting a vote" does not depend on the structure of the deleted vote. Here, it is interesting whether some of the positive results, such as fixed-parameter tractability with respect to the "number of candidates", transfer.

  Above we discussed the modification of voting systems such that they select a winner also for partial orders. Another way to deal with incomplete information is to consider whether the given partial orders can be extended such that a distinguished candidate wins. This leads to the Possible Winner problem which is studied in the second part of this work for "easy-to-evaluate" voting rules (also see Chapter 10 for open questions in this direction).

- Regarding kernelization for Dodgson Score with respect to the "number of switches", the existence of a polynomial-size problem kernel is very unlikely [106]. However, the problem might still allow for an efficient partial kernelization as introduced in Chapter 4. Furthermore, note that the exponential-size kernel for a generalized version of Dodgson Score [108] does not imply a kernelization for Dodgson Score itself. Hence, the development of (practically effective) data reduction rules for Dodgson Score leading to any provable performance guarantee is an interesting task.

  For Young Score, kernelization results with respect to the two standard voting parameters would clearly be of interest.

- Bartholdi et al. [12] gave an integer linear program which implies the fixed-parameter tractability of Dodgson Score with respect to the parameter "number of candidates" (also see [163] for further results in this direction). Unfortunately, the corresponding running times are extremely high and a more efficient combinatorial algorithm would be desirable; the same holds true for Young Score.

- Another view on Dodgson's election system is provided by the so-called "distance rationalizability" framework [8, 170] in which a voting rule is defined based on a class of "consensus elections" and a distance function measuring the distance to this class. For Dodgson's rule, the class of consensus elections comprises all election having a Condorcet winner and the distance function is given by the number of switches. This framework invites for studies of classes of voting systems. A first fixed-parameter tractability result with respect to the "number of candidates" for a broad class of voting rules has been recently provided by Elkind et al. [82].

Part II

Possible Winner Determination

To make a joint decision, in the standard model of elections, voters are often required to provide their preferences as linear orders. To determine a winner, the given linear orders are aggregated according to a voting protocol. However, in realistic settings, the voters may often only provide partial orders. This directly leads to the Possible Winner problem that asks, given a set of partial votes, whether a distinguished candidate can still become a winner. In this part of the thesis, we consider the computational complexity of Possible Winner for scoring rules from various perspectives. This part comprises three chapters. In Chapter 7, we investigate the influence of the type of the scoring rule on the computational complexity. In Chapter 8, we investigate the parameterized complexity of Possible Winner under some scoring rules with respect to several single parameterizations. Chapter 9 is concerned with combined parameterizations for the k-approval voting system. Finally, Chapter 10 concludes this part of the thesis.

Chapter 7

A dichotomy for pure scoring rules

Scoring rules form a broad class of voting protocols including many well-known rules like plurality, k-approval, or Borda. A natural question is how the "kind of scoring rule" influences the computational complexity of the Possible Winner problem. A multivariate complexity analysis provides possible ways to address this question. Examples for natural parameterizations comprise "the number of candidates getting more than zero points per vote" or "the maximum number of candidates getting an equal/different number of points per vote". We provide evidence that it is very unlikely that such parameterizations lead to tractable cases. Generalizing previous NP-hardness results for some special cases, we settle the computational complexity of Possible Winner for all but one scoring rule. More precisely, for an unbounded number of candidates and unweighted voters, we show that Possible Winner is NP-complete for all pure scoring rules except plurality, veto, and the scoring rule defined by the scoring vector $(2, 1, \ldots, 1, 0)$, while it is solvable in polynomial time for plurality and veto. The remaining case $(2, 1, \ldots, 1, 0)$ has been shown NP-complete in a follow-up work by Baumeister and Rothe [14], yielding a full dichotomy.

7.1 Motivation and known results

Voting scenarios arise whenever the preferences of different parties have to be aggregated to form a joint decision. Often, the voting process is executed in the following way: each voter provides his preference as a linear order of all the possible alternatives/candidates. However, in realistic settings, the voters may often only provide partial orders (or partial votes) instead of linear ones. For example, it might be impossible for the voters to provide a complete preference list because the set of candidates is too large, as it is the case for web page ranking. Another reason might be that not all voters might have given their preferences yet during the aggregation process, or new candidates might be introduced after some voters already have given their rankings. Moreover, one often has to deal with partial votes due to incomparabilities: for some voters it might not be possible to compare two candidates or certain groups of candidates, be it because of lack of information or due to personal reasons. Hence, the study of partial voting profiles is natural and essential. One question that immediately

comes to mind is whether any information on a possible outcome of the voting process can be given in the case of incomplete votes. More specifically, in this part of the thesis, we study the Possible Winner problem: Given a partial order for each of the voters, can a distinguished candidate c win for at least one extension of the partial orders into linear ones?

Of course, the answer to this question depends on the voting rule that is used. In this chapter, we will stick to the broad class of voting protocols defined by *scoring rules* [38]. A scoring rule provides a score value for every position that a candidate can take within a linear order, given as a *scoring vector* of length m in the case of m candidates. The scores of the candidates are then added over all votes and the candidates with the highest score win. Famous examples are Borda, defined by the scoring vectors $(m-1, m-2, \ldots, 0)$ and k-approval, defined by $(1, \ldots, 1, 0, \ldots, 0)$ starting with k ones. Two relevant special cases of k-approval are plurality, defined by $(1, 0, \ldots, 0)$, and veto, defined by $(1, \ldots, 1, 0)$. Typically, k-approval can be used in political elections whenever the voters can express their preference for k candidates within the set of all candidates. Another example is the Formula 1 scoring, which until the year 2009 used the scoring rule defined by the vector $(10, 8, 6, 5, 4, 3, 2, 1, 0, \ldots, 0)$ and since 2010 uses $(25, 18, 15, 12, 10, 8, 6, 4, 2, 1, 0, \ldots, 0)$.

The Possible Winner problem was introduced by Konczak and Lang [148] and has been further investigated since then for many types of voting systems [14, 28, 54, 153, 177, 193, 194]. Note that the related Necessary Winner problem (Given a set of partial orders, does a distinguished candidate c win for every extension of the partial orders into linear ones?) can be solved in polynomial time for all scoring rules [194], and is hence not considered in this work. We summarize the known results for Possible Winner and its prominent special case Manipulation for scoring rules. For scoring rules that are defined for a constant number of candidates, the Possible Winner problem can be decided in polynomial time, see [65, 193]. Otherwise, correcting Konczak and Lang [148] who claimed polynomial-time solvability for all scoring rules, Xia and Conitzer [194] provided NP-completeness results for a class of scoring rules, more specifically, for all scoring rules that have four "equally decreasing score values" followed by another "strictly decreasing score value"; we will provide a more detailed discussion in Section 7.5. In addition, all NP-hardness results directly carry over from Manipulation to Possible Winner. However, whereas the case of *weighted voters* is settled by a full dichotomy for Manipulation for scoring rules [129], so far, for *unweighted voters* we are only aware of one NP-hardness result for a specially constructed scoring rule [199].

Altogether, until now, the computational complexity of Possible Winner was still open for a large number of naturally appearing scoring rules. One such open case has been k-approval for small values of k motivated as follows. A common way of voting for a board consisting of a small number, for example, of five members, is that every voter awards five candidates one points each time (5-approval). A second example is given by voting systems in which each voter is allowed to specify a (small) group of favorites and a (small) group of most disliked candidates. As final example, we mention scoring rules that have decreasing differences between successive score values as, for example, the scoring vector $(2^m, 2^{m-1}, \ldots, 0)$.

This chapter aims at showing a computational complexity dichotomy for *pure* scoring rules. The class of pure scoring rules covers all of the common scoring rules. It only constitutes some restrictions in the sense that for different numbers of candidates

the corresponding scoring vectors can not be chosen completely independently (see Section 7.2). Our results can also be extended to broad classes of "non-pure" scoring rules, cf. Section 7.7. Altogether, we settle the computational complexity of Possible Winner for all pure scoring rules except the scoring rule defined by $(2, 1, \ldots, 1, 0)$. For plurality and veto, we provide polynomial-time algorithms whereas for the remaining cases we show NP-completeness. Surprisingly, this includes the NP-hardness of Possible Winner even for 2-approval. Our NP-hardness result for 2-approval has also been used to settle the complexity of the Swap Bribery problem [80].

The technical contributions of this chapter are the following. We identify a strategy allowing for a classification of all scoring rules into (at least one of) two types. We distinguish between scoring rules with an "unbounded number of different score values" and scoring rules with an "unbounded number of equal score values". For the first type of scoring rules, we give a gadget construction that extends and generalizes the Exact Cover By 3-Sets-reduction due to Xia and Conitzer [194]. For the second type of scoring rules, our NP-hardness results are based on new many-one reductions: For k-approval and related/generalized scoring rules, we provide reductions from Multicolored Clique. Besides some reductions to settle more special cases, we give another family of reductions from Exact Cover By 3-Sets for all but one of the scoring rules allowing that one can specify one favorite and one most disliked candidate. Note that this reduction is based on an approach different from the reduction for the first type of scoring rules.

7.2 Definitions

(Positional) scoring rules are a special kind of voting rules. They are defined by scoring vectors $\overrightarrow{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ with integers $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_m$, the *score values*. More specifically, we define that a scoring rule r consists of a sequence of scoring vectors $s_1, s_2, \ldots$ such that for any $i \in \mathbb{N}_{>0}$ there is a scoring vector s_i for i candidates which can be computed in time polynomial in i. Famous examples are

- Borda defined by $(m-1, m-2, \ldots, 1, 0)$ for m candidates and
- k-approval defined by $(1, \ldots, 1, 0, \ldots, 0)$ starting with k ones.

For a k-approval scoring vector, we denote the first k positions as *one-positions* and the remaining positions as *zero-positions*. On the one side, k-approval generalizes *plurality* where every voter gives one point, and, on the other side, it extends *veto* where every voter gives one point to all but one candidate. The veto scenario is useful whenever one needs to "exclude" few alternatives, for example, decreasing the number of postal offices by closing few of them.

In this chapter, we focus our attention on *pure* scoring rules, that is for every $i \geq 2$, the scoring vector for i candidates can be obtained from the scoring vector for $i-1$ candidates by inserting an additional score value at an arbitrary position (respecting that the score values must be monotonously decreasing). This definition includes all of the common protocols. We further assume that $\alpha_m = 0$ and that there is no integer greater than one that divides all score values. This does not constitute a restriction since for every other voting system there must be an equivalent one that fulfills these constraints [129, Observation 2.2]. Moreover, we only consider *non-trivial*

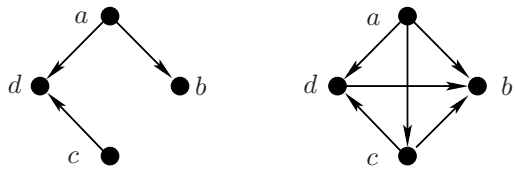


Figure 7.1: Partial vote $a \succ \{b, d\}$, $c \succ d$ (left-hand side) and one of its possible extensions $a > c > d > b$ (right-hand side). An arc from a to b means that a is preferred to b.

$$\begin{array}{lcl} a \succ \{b, c, d\} & \Rightarrow & a > c > d > b \\ a \succ \{b, c, d\} & \Rightarrow & a > c > d > b \\ a \succ \{b, c, d\},\ b \succ d & \Rightarrow & a > c > b > d \\ a \succ b,\ c \succ d & \Rightarrow & c > d > a > b \end{array}$$

Figure 7.2: Four partial votes on the candidate set $\{a, b, c, d\}$ (left-hand side) and a winning extension for 2-approval (right-hand side) with distinguished candidate c. In contrast, the candidate c cannot be a possible winner for plurality since clearly a wins in every extension of the four partial votes.

scoring rules, that is, scoring rules with $\alpha_1 \neq 0$ for scoring vectors for every number of candidates.

Consider a profile P on a set C of candidates. For a vote $v \in P$ and a candidate $c \in C$, let the *score* $s(v, c)$ be defined by $s(v, c) := \alpha_j$ where j is the position of c in v. For any profile $P = \{v_1, \ldots, v_n\}$, let $s(P, c) := \sum_{i=1}^{n} s(v_i, c)$. Whenever it is clear from the context which P we refer to, we will just write $s(c)$. A scoring rule selects all candidates c as winners with maximum $s(P, c)$ over all candidates.

A *partial vote* on C is a transitive and antisymmetric relation on C, see Figure 7.1 for an example. We use $>$ to denote the relation given between candidates in a linear order and $\succ$ to denote the relation given between candidates in a partial vote. We omit the relative order of candidate pairs if the order follows directly by transitivity from other pairs. Sometimes, we specify a whole subset of candidates in a partial vote, e.g., $e \succ D$ for a candidate $e \in C$ and a subset of candidates $D \subseteq C$. Unless stated otherwise, this notation means that $e \succ d$ for all $d \in D$ and there is no specified order among the candidates in D. In contrast, writing $e > D$ in a linear order means that $e > d_1 > \cdots > d_l$ for an arbitrary but fixed order of $D = \{d_1, \ldots, d_l\}$. A linear order v' *extends* a partial vote v if $v \subseteq v'$, that is, for all $i, j \leq m$, from $c_i \succ c_j$ in v it follows that $c_i > c_j$ in v'. Given a profile of partial votes $P = (v_1, \ldots, v_n)$ on C, a candidate $c \in C$ is a *possible winner* if there exists an extension $P' = (v'_1, \ldots, v'_n)$ such that each v'_i extends v_i and c wins according to a specified voting rule. See Figure 7.2 for an example. For any voting rule r, the corresponding decision problem is as follows.

Possible Winner
Given: A set of candidates C, a profile of partial votes $P = (v_1, \ldots, v_n)$ on C, and a distinguished candidate $c \in C$.
Question: Is there an extension profile $P' = (v'_1, \ldots, v'_n)$ such that each v'_i extends v_i and $c \in r(P')$?

Table 7.1: Overview of results and outline of the work. Basically, we partition the scoring rules into five different types according to the types of algorithms or many-one reductions that are used to achieve the results. By "different-type" we denote all scoring vectors with an unbounded number of different score values. By "equal-type" we denote all scoring vectors with an unbounded number of equal score values if not listed explicitly in another type. Reductions are from Exact Cover By 3-Sets (X3C) or Multicolored Clique (MC).

Scoring rule	Result	
Plurality and Veto	in P	Proposition 7.1, Section 7.4
different-type	NP-c (X3C)	Theorem 7.1, Section 7.5
equal-type	NP-c (MC/X3C)	Theorem 7.2, Lemmata 7.3 – 7.6, Section 7.6.1
$\alpha_1 > \alpha_2 = \alpha_{m-1} > 0$ and $\alpha_1 \neq 2 \cdot \alpha_2$	NP-c (X3C)	Theorem 7.4, Section 7.6.2
$(2, 1, \ldots, 1, 0)$	NP-c	[14]

This definition allows that multiple candidates obtain the maximal score and we end up with a whole set of winners. If the possible winner c has to be unique, one speaks of a *possible unique winner*, and the corresponding decision problem is defined analogously. All our results hold for both cases (see Section 7.3.1).

In this and the following chapter, several of our NP-hardness proofs rely on reductions from the NP-complete Exact Cover By 3-Sets (X3C) problem [118] defined as follows.

> *Given:* A set of elements $E = \{e_1, \ldots, e_q\}$, a family of subsets $\mathcal{S} = \{S_1, \ldots, S_t\}$ with $|S_i| = 3$ and $S_i \subseteq E$ for $1 \leq i \leq t$.
> *Question:* Is there a subset $\mathcal{S}' \subseteq \mathcal{S}$ such that for every element $e_j \in E$ there is exactly one $S_i \in \mathcal{S}'$ with $e_j \in S_i$?

In some of our theorems, we will need functions that map each instance of a certain problem $\mathcal{Q}$ to some natural number and in some sense behave like a polynomial. For this sake, we call

$$f : \{I \mid I \text{ is an instance of } \mathcal{Q}\} \to \mathbb{N}$$

a *poly-type function for* $\mathcal{Q}$ if the function value $f(I)$ is bounded by a polynomial in $|I|$ for every input instance I of $\mathcal{Q}$.

7.3 General strategy

This chapter aims at providing a dichotomy for Possible Winner for practically relevant scoring rules. To this end, we will show the following.

Theorem. Possible Winner *is NP-complete for all non-trivial pure scoring rules except plurality, veto, and scoring rules for which there is a constant z such that the produced scoring vector is* $(2, 1, \ldots, 1, 0)$ *for every number of candidates greater than z. For plurality and veto,* Possible Winner *is solvable in polynomial time.*

The proof consists of several parts, see Table 7.1 for an overview. The polynomial time results for plurality and veto are based on flow computations. Regarding the NP-hardness results, we give many-one reductions that work for scoring rules that produce specific "types of scoring vectors" for an appropriate number of candidates. We combine the single results to obtain the main result in Section 7.7. To this end, we have to take into account that, in general, a scoring rule might produce different types of scoring vectors for different numbers of candidates.

The basic observation to classify the scoring vectors is that a scoring vector of unbounded size must have an unbounded number of different score values or an unbounded number of equal score values. This leads to the following strategy. First, we show NP-hardness for all scoring vectors having an unbounded number of different score values. To this end, we generalize a many-one reduction due to Xia and Conitzer [194]. Second, we deal with scoring vectors having an unbounded number of equal score values. Here, we consider two subcases, i.e., scoring vectors of type $\alpha_1 > \alpha_2 = \alpha_{m-1} > 0$ but $\alpha_1 \neq 2 \cdot \alpha_2$, and all remaining scoring vectors with an unbounded number of equal score values.

Before stating the specific results, we give a construction scheme that is used in all many-one reductions in this work.

7.3.1 A general scheme to construct linear votes

In all many-one reductions presented in this work, one constructs a partial profile P consisting of a set of linear orders V^l and a set of partial votes V^p. The position of the distinguished candidate c is already determined in every vote from V^p, that is, $s(P', c)$ is the same in every extension P' and thus is fixed. The "interesting" part of the reductions is given by the partial votes of V^p in combination with upper bounds for the scores which the non-distinguished candidates can make in V^p. For every candidate $c' \in C \backslash \{c\}$, the *maximum partial score* $s_p^{\max}(c')$ is the maximum number of points c' may make in V^p without beating c in P. More precisely, for the unique winner case, $s_p^{\max}(c') = s(P', c) - s(V^l, c') - 1$ and, for the winner case, $s_p^{\max}(c') = s(P', c) - s(V^l, c')$ for any extension P' of P. Since the maximum partial scores can be adjusted to the unique and to the winner case, all results hold for both cases.

In the following, we show that for all our reductions, there is an easy way to cast the linear votes such that the maximum partial scores that are required in the reductions are *realized*. For every many-one reduction of this work, it will be easy to verify that the underlying partial profile fulfills the following two properties.[1]

> **Property 1** There is a "dummy" candidate d which cannot beat the distinguished candidate in any extension, that is, $s_p^{\max}(d) \geq \alpha_1 \cdot |V^p|$.
>
> **Property 2** For every $c' \in C \backslash \{c\}$, the maximum partial score $s_p^{\max}(c')$ can be written as a sum of at most $|V^p|$ integers from $\{\alpha_1, \ldots, \alpha_m\}$. Formally, the definition of $s_p^{\max}(c')$ will be of the form $s_p^{\max}(c') = \sum_{j=1}^{m} n_j \, \alpha_j$ where $n_j \in \mathbb{N}_0$ denotes how often the score value α_j is added. We will always have that $\sum_{j=1}^{m} n_j \leq |V^p|$, that is, the total number of summands is at most the number of partial votes.

[1] The only exception appears in the proof of Theorem 7.4 and will be discussed there.

$$
\begin{array}{lccccccccc}
v_1: & c_1 & > & c_2 & > & \dots & > & c_{m-1} & > & c_m \\
v_2: & c_2 & > & c_3 & > & \dots & > & c_m & > & c_1 \\
\vdots & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
v_{m-1}: & c_{m-1} & > & c_m & > & \dots & > & c_{m-3} & > & c_{m-2} \\
v_m: & c_m & > & c_1 & > & \dots & > & c_{m-2} & > & c_{m-1}
\end{array}
$$

Figure 7.3: Circular block for $c_1, c_2 \dots, c_m$.

The sets of linear votes which are necessary for the reductions given in this paper can be obtained according to the following lemma.

Lemma 7.1. *Given a scoring rule r, a set of candidates C with distinguished candidate $c \in C$, a set of partial votes V^p in which c is fixed, and $s_p^{\max}(c')$ for all $c' \in C \backslash \{c\}$, a set of linear votes that realizes the maximum partial scores for all candidates can be constructed in time polynomial in $|V^p|$ and m if Properties 1 and 2 hold.*

Proof. We are interested in "setting" relative score difference between the distinguished candidate c and every other candidate. By inserting one linear order we change the relative score difference between c and all other candidates. To be able to change the relative score difference only for c and one specific candidate while keeping the relative score difference of c and all other candidates, we will build V^l by sets of circular shifts instead of single votes. More precisely, for a set of candidates $\{c_1, c_2, \dots, c_m\}$, a *circular block* consists of m linear orders as given in Figure 7.3. Clearly, all candidates have the same score within a circular block.

We start with the construction for the winner case and then explain how to adapt it for the unique winner case. For the winner case ($s_p^{\max}(c') = s(P', c) - s(V^l, c')$ for any extension P'), for each candidate $c' \in C \backslash \{c, d\}$ where d denotes a dummy as specified in Property 1, add the following votes to the set of linear votes V^l. For each $n_j \neq 0$ as specified in Property 2, construct n_j circular blocks over C such that in one of the linear orders of every block, c' sits on position j and d sits on position m. Exchange the places of c' and d in this linear order and add the modified circular block to V^l. Then, for one block, c' has lost α_j points and gained $\alpha_m = 0$ points relative to c. Thus, in total, one has the situation that c and c' have exactly the same score if c' makes $s_p^{\max}(c')$ points in V^p. This settles the winner case. For the unique-winner case, we additionally decrease the score of c' by the minimum of $\{\alpha_i - \alpha_j \mid \alpha_i > \alpha_j \text{ and } i, j \in \{1, 2, \dots, m\}\}$. This can be achieved by adding a circular block such that in one of the linear orders of the block, c' sits on position α_i and d sits on position α_j, and by exchanging the places of c' and d in this linear order. Then, c beats c' if c' makes at most $s_p^{\max}(c')$ points in V^p and c' beats c, otherwise.

Altogether, due to Property 2, we add at most $|V^p|$ summands for each candidate. Hence, so far, the number of linear votes is bounded by $m^2 \cdot (|V^p| + 1)$ and can be constructed in polynomial time. It remains to adjust the maximum partial score of d. Until now, we added at most $m \cdot (|V^p| + 1)$ circular blocks. Thus, d can make at most $\alpha_1 \cdot m \cdot |V^p|$ points more than c. By adding $m(|V^p| + 1) + |V^p|$ further circular blocks for candidates from $C \backslash \{d\}$ that are inserted in the first $m - 1$ positions, while d is put on the last position in these votes, $s_p^{\max}(d)$ can be realized in polynomial time. □

$v_1 : a \succ c \succ d, b \succ c$
$v_2 : c \succ a \succ b \quad \Rightarrow c > a > b > d$
$v_3 : a \succ d \succ b \quad \Rightarrow c > a > d > b$
$v_4 : a \succ b \succ c$
$v_5 : a \succ c, b \succ d$

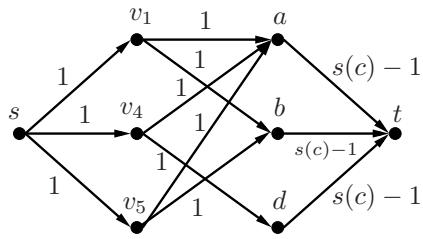


Figure 7.4: Possible Winner for plurality: The left-hand side shows an example for an election and the right-hand side the corresponding flow network. The votes v_2 and v_3 can be extended such that c takes the first position. The position of the remaining candidates in theses votes is not relevant; one possibility how to extend these votes is shown in the picture.

7.4 Plurality and veto

Employing network flows turned out to be useful to design algorithms for several voting problems (see e.g. [93, 99]). Here, by using some flow computations very similar to [28, Theorem 6], we show the following.

Proposition 7.1. Possible Winner *can be solved in polynomial time for plurality and veto.*

Proof. First, we give an algorithm for plurality. Let P on C denote a Possible Winner-instance with distinguished candidate c. Clearly, it is safe to set c to the first position in all votes in which this is possible. Then the score of c is fixed at the maximum possible value. We denote the partial votes of P in which the first position is not taken by c as P_1. Now, we can model the problem as network flow as follows (see Figure 7.4): The flow network consists of a source node s, a target node t, one node for every vote of P_1, and one node for every candidate from $C \backslash \{c\}$. There are three layers of arcs:

1. an arc from s to every node corresponding to a vote in P_1 with capacity one,
2. an arc from a node corresponding to $v_j \in P_1$ to a node corresponding to a candidate $c' \in C \backslash \{c\}$ with capacity one if and only if c' can take the first position in an extension of v_j, and
3. an arc from every node corresponding to $c' \in C \backslash \{c\}$ to target t with capacity $s(c) - 1$.

Now, c is a possible winner if and only if there is a flow of size $|P_1|$: The first layer simulates that the first position of every partial vote from P_1 has to be taken, the second layer that it can only be taken by appropriate candidates, and the last one that the score of every candidate will be lower than the score of c. Clearly, the flow network can be constructed in time polynomial in $|P_1|$ and an integral flow computation can be done in polynomial time [66].

For veto, we first fix c at the best (leftmost) possible position in every vote. This fixes the maximum score of c. Then for every candidate $c' \in C \setminus \{c\}$, let $z(c')$ denote

the minimum number of last positions that c' must take such that it does not beat c. Let P_1 denote the set of partial votes in which c does not take the last position. Again, we model the problem by a flow network with source node s, target node t, one node for every candidate from $C\backslash\{c\}$, and one node for every vote of P_1. The arcs are as follows:

1. an arc from s to every node corresponding to $c' \in C\backslash\{c\}$ with capacity $z(c')$,
2. an arc from a node corresponding to $c' \in C\backslash\{c\}$ to a node corresponding to $v_j \in P_1$ with capacity one if and only if c' can take the last position in an extension of v_j, and
3. an arc from every node corresponding to $v_j \in P_1$ to target t with capacity 1.

By similar arguments as for plurality, it follows that c is a possible winner if and only if there is a flow of size $\sum_{c' \in C\backslash\{c\}} z(c')$. □

7.5 An unbounded number of positions with different score values

Xia and Conitzer [194] developed a many-one reduction from Exact Cover By 3-Sets showing that Possible Winner is NP-complete for any scoring rule with scoring vectors which contain four consecutive, "equally decreasing" score values, followed by another strictly decreasing score value. Using some additional gadgetry, we extend their proof to work for scoring vectors with an unbounded number of different, not necessarily equally decreasing score values.

We start by describing the basic idea employed in [194] (using a slightly modified construction). Given an X3C-instance $(E, \mathcal{S})$, construct a partial profile $P := V^l \cup V^p$ on a set of candidates C where V^l denotes a set of linear orders and V^p a set of partial votes. To describe the basic idea, assume that there is a scoring vector with $\alpha_1 > \alpha_2$ and and the differences between the four following score values are *equally decreasing*, that is, $\alpha_2 - \alpha_3 = \alpha_3 - \alpha_4 = \alpha_4 - \alpha_5$. Then, $C := \{c, x, w\} \cup E$ where E is the universe from the X3C-instance. The distinguished candidate is c. The candidates whose element counterparts belong to the set S_i are denoted by e_{i1}, e_{i2}, e_{i3}. The partial votes V^p consist of one partial vote v_i^p for every $S_i \in \mathcal{S}$ which is given by

$$x \succ e_{i1} \succ e_{i2} \succ e_{i3} \succ C',\ w \succ C'$$

with $C' := C\backslash\{x, e_{i1}, e_{i2}, e_{i3}, w\}$. Note that in v_i^p, the positions of all candidates except $w, x, e_{i1}, e_{i2}, e_{i3}$ are fixed. More precisely, w has to be inserted between positions 1 and 5 maintaining the partial order $x \succ e_{i1} \succ e_{i2} \succ e_{i3}$. By setting the linear votes, the maximum partial scores are realized such that the following three conditions hold.

- For every *element candidate* $e \in E$ one has the following. Inserting w behind e in two partial votes has the effect that e would beat c, whereas when w is inserted behind e in at most one partial vote, c still beats e (Condition 1). Note that e may occur in several votes at different positions, e.g. e might be identical with e_{i1} and e_{j3} for $i \neq j$. However, due to the condition of "equally decreasing" scores, "shifting" e increases its score by the same value in all of the votes.

- The maximum partial score of x is set such that if takes more than $|V^p| - |E|/3$ times the first position, then it would beat c. That is, w must be inserted before x at least $|V^p| - |E|/3$ times (Condition 2).
- We set $s_p^{\max}(w) = (|V^p| - |E|/3) \cdot \alpha_1 + |E|/3 \cdot \alpha_5$. This implies that if w is inserted before x in $|V^p| - |E|/3$ votes, then it must be inserted at the last possible position, that is, position 5, in all remaining votes (Condition 3).

Having an exact 3-cover for $(E, \mathcal{S})$, extend the partial votes as follows.

$v_i^p : x > e_{i1} > e_{i2} > e_{i3} > w > \ldots$ if S_i is in the exact 3-cover
$v_i^p : w > x > e_{i1} > e_{i2} > e_{i3} > \ldots$ if S_i is not in the exact 3-cover.

Then, every element candidate e is shifted exactly once (in v_i^p for $e \in S_i$, if S_i is in the exact 3-cover) and thus is beaten by c. It is easy to verify that c beats w and x as well. In a yes-instance for (C, P, c), it follows directly from Condition 2 and 3 that w must have the position 5 in exactly $|E|/3$ votes and the first position in all remaining partial votes. Since there are $|E|/3$ partial votes such that three element candidates are shifted in each of them, due to Condition 1, every element candidate must appear in exactly one of these votes. Hence, c is a possible winner in P if and only if there exists an exact 3-cover of E.

By inserting further candidates, one can pad the construction such that is also works if the equally decreasing score differences appear at other positions [194]. Now, we consider the situation in which no such equally decreasing score differences appear at all. More precisely, we show how to extend the reduction to scoring vectors with strictly, but not equally decreasing scoring values. The problem we encounter is the following: By sending candidate w to the last possible position in the partial vote v_i^p, each of the candidates e_{i1}, e_{i2}, e_{i3} improves by one position and therefore improves its score by the difference given between the corresponding positions. In [194], these differences all had the same value, but now we have to deal with varying differences. Since the same candidate $e \in E$ may appear in several votes at different positions, e.g. e might be identical with e_{i1} and e_{j3} for $i \neq j$, it is not clear how to set the maximum partial score of e. Basically, to cope with this situation, we construct three partial votes v_i^1, v_i^2, and v_i^3 for every set $S_i \in \mathcal{S}$ and permute the positions of the candidates e_{i1}, e_{i2}, e_{i3} such that each of them takes a different position in v_i^1, v_i^2, v_i^3. For example:

$$v_i^1 : \ldots \succ x \succ e_{i1} \succ e_{i2} \succ e_{i3} \succ \ldots$$
$$v_i^2 : \ldots \succ x \succ e_{i2} \succ e_{i3} \succ e_{i1} \succ \ldots$$
$$v_i^3 : \ldots \succ x \succ e_{i3} \succ e_{i1} \succ e_{i2} \succ \ldots$$

In this way, if the candidate w is sent to the last possible position in all three partial votes of a set S_i, each of the candidates e_{i1}, e_{i2}, e_{i3} improves its score by the same value. We only have to guarantee that whenever w is sent back in the partial vote v_i^1, then it has to be sent back v_i^2 and v_i^3 as well. This is realized by a gadget construction, which is the main technical contribution of the following theorem.

Theorem 7.1. *An X3C-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a scoring vector having $f(I)$ positions with different score values. A suitable poly-type function f can be computed in polynomial time.*

Table 7.2: Maximum partial scores. Recall that $t = |\mathcal{S}|$, $q = |E|$, and $n_e = |\{S_i \in \mathcal{S} \mid e \in S_i\}|$.

	$s_p^{\max}(w)$	$= (3t-q)\cdot\alpha_1 + q\cdot\alpha_{5+2t}$
	$s_p^{\max}(x)$	$= q\cdot\alpha_1 + (3t-q)\cdot\alpha_2$
$\forall e \in E$	$s_p^{\max}(e)$	$= (\alpha_2+\alpha_3+\alpha_4) + (n_e-1)\cdot(\alpha_3+\alpha_4+\alpha_5) + \text{fixed}(e)$
$\forall d_i$	$s_p^{\max}(d_i)$	$= q/3\cdot\alpha_{4+i} + (t-q/3)\cdot\alpha_{5+i} + \text{fixed}(d_i)$
$\forall h_i$	$s_p^{\max}(h_i)$	$= q/3\cdot\alpha_{4+i} + (t-q/3)\cdot\alpha_{5+i} + \text{fixed}(h_i)$
$\forall d'_i$	$s_p^{\max}(d'_i)$	$= q/3\cdot\alpha_{4+i+t} + (t-q/3)\cdot\alpha_{5+i+t} + \text{fixed}(d'_i)$
$\forall h'_i$	$s_p^{\max}(h'_i)$	$= q/3\cdot\alpha_{4+i+t} + (t-q/3)\cdot\alpha_{5+i+t} + \text{fixed}(h'_i)$
$\forall l_i$	$s_p^{\max}(l_i)$	$= 2t\cdot\alpha_1 + \text{fixed}(l_i)$

Proof. Given an X3C-instance $(E, \mathcal{S})$ with $\mathcal{S} = \{S_1, \ldots, S_t\}$ and $S_i = \{e_{i1}, e_{i2}, e_{i3}\}$ for $i \in \{1, \ldots, t\}$, construct a partial profile P on C as follows. The set of candidates is defined as $C := \{x, w, c\} \uplus E \uplus D_{12} \uplus D_{13} \uplus L$ (where $\uplus$ denotes the disjoint union), where E is the set of candidates that represent the elements of the universe of the X3C-instance, $D_{12} := \{d_1, \ldots, d_t, h_1, \ldots, h_t\}$, $D_{13} := \{d'_1, \ldots, d'_t, h'_1, \ldots, h'_t\}$, and $L := \{l_1, \ldots, l_t\}$. We define $f((E, \mathcal{S})) := |C|$. To ease the presentation, we first assume that we have a strictly decreasing scoring vector of size $f((E, \mathcal{S}))$ and describe how to generalize this at the end of the proof. The partial profile consists of a set of partial votes V^p and a set of linear votes V^l. The partial votes are $V^p := \{v_i^1, v_i^2, v_i^3 \mid 1 \leq i \leq t\}$ with,

for $1 \leq i \leq t-1$,

$v_i^1 : x\ e_{i1}\ e_{i2}\ e_{i3}\ d_1 \ldots d_i\ h_{i+1} \ldots h_t\ d'_1 \ldots d'_i\ h'_{i+1} \ldots h'_t \succ C_i^1\ ,\ w \succ C_i^1$

$v_i^2 : x\ e_{i2}\ e_{i3}\ e_{i1}\ h_1 \ldots h_i\ d_{i+1} \ldots d_t\ l_1 \ldots \quad \ldots \ldots l_t \succ C_i^2\ ,\ w \succ C_i^2$

$v_i^3 : x\ e_{i3}\ e_{i1}\ e_{i2}\ l_1 \ldots \quad \ldots \ldots l_t\ h'_1 \ldots h'_i\ d'_{i+1} \ldots d'_t \succ C_i^3\ ,\ w \succ C_i^3$

and

$v_t^1 : x\ e_{t1}\ e_{t2}\ e_{t3}\ d_1 \ldots d_t\ d'_1 \ldots d'_t \succ C_t^1\ ,\ w \succ C_t^1$

$v_t^2 : x\ e_{t2}\ e_{t3}\ e_{t1}\ h_1 \ldots h_t\ l_1 \ldots l_t \succ C_t^2\ ,\ w \succ C_t^2$

$v_t^3 : x\ e_{t3}\ e_{t1}\ e_{t2}\ l_1 \ldots l_t\ h'_1 \ldots h'_t \succ C_t^3\ ,\ w \succ C_t^3$

where "$\succ$" signs are partially omitted and C_i^1, C_i^2, and C_i^3 denote the remaining candidates that are fixed in an arbitrary order, respectively. Now, we give some notation needed to define the maximum partial scores. For $c' \in C \backslash \{c\}$, let fixed$(c')$ denote the number of points which c' makes in the partial votes in which the position of c' is already fixed. Let n_e denote the number of subsets with $e \in S_i$ and $q = |E|$. Due to Lemma 7.1, one can set the maximum partial scores as given in Table 7.2. The particular partial scores will be explained within the proof of the following claim.

Claim: Candidate c is a possible winner of P if and only if there is an exact 3-cover for $(E, \mathcal{S})$.

"$\Leftarrow$": Given an exact 3-cover $\mathcal{S}' \subseteq \mathcal{S}$, complete the votes in V^p in the following way: For each $S_i \in \mathcal{S}'$, place w in the last possible position (i.e., position $5+2t$) in the partial votes v_i^1, v_i^2, and v_i^3, and on the first position in the remaining partial votes. Since $|\mathcal{S}'| = q/3$, in the extension of the votes from V^p ones has $s(w) = (3t-q)\cdot\alpha_1 + q\cdot\alpha_{5+2t} = s_p^{\max}(w)$ and $s(x) = q\cdot\alpha_1 + (3t-q)\cdot\alpha_2 = s_p^{\max}(x)$. Furthermore, it is easy to see that $s(l_i) < s_p^{\max}(l_i)$ for every i. Every element candidate e is shifted to the left in

exactly three partial votes. More precisely, in the three votes that correspond to $S_i \in \mathcal{S}'$ with $e \in S_i$, it makes $\alpha_2 + \alpha_3 + \alpha_4$ points and $(n_e - 1) \cdot (\alpha_3 + \alpha_4 + \alpha_5) + \text{fixed}(e)$ points in the remaining votes and thus does not beat c. Every candidate from D_{12} is not "fixed" in exactly one vote of every triple corresponding to an S_i. More precisely, it can be shifted either in v_i^1 or in v_i^2 and never in v_i^3. Due to the insertion of w, it is shifted to position $4 + i$ in $q/3$ of the votes and takes position $5 + i$ in the remaining $t - q/3$ non-fixed votes. Thus, it does not beat c. Analogously, every candidate from D_{13} makes α_{4+i+t} points in $q/3$ of the non-fixed votes and α_{5+i+t} in the remaining $t - q/3$ votes and hence does not beat c. Altogether, c beats every other candidate and wins.

"$\Rightarrow$": Consider an extension of P in which c wins. Due to its maximum partial score, candidate x can take the first position only q times. Thus, it must be shifted $3t - q$ times to position 2. Clearly, this is only possible if w is placed on the first position in $3t - q$ votes. Then due to its maximum partial score, w can only be set to position $5 + 2t$ in the remaining q votes. In the following, we will show that for every i, w takes position $5 + 2t$ in v_i^1 if and only if it takes position $5 + 2t$ in v_i^2 if and only if it takes position $5 + 2t$ in v_i^3 (Observation I). Then it follows that in the votes in which w takes position $5 + 2t$, the corresponding element candidates are shifted to the left and obtain $\alpha_2 + \alpha_3 + \alpha_4$ points each, whereas they obtain $\alpha_3 + \alpha_4 + \alpha_5$ points in the remaining corresponding vote triples. Since each element candidate e_j can only obtain $\alpha_2 + \alpha_3 + \alpha_4$ points exactly once (and the scoring values are strictly decreasing), the set $\mathcal{S}' := \{S_i \mid w \text{ is placed on position } 5 + 2t \text{ in } v_i^1\}$ must be an exact 3-cover of E.

It remains to settle Observation I, which says that w behaves equally in the votes corresponding to one subset. First, we argue that w must be inserted at position $5+2t$ in exactly $q/3$ votes of $V_1^p := \{v_i^1 \mid 1 \leq i \leq t\}$, $V_2^p := \{v_i^2 \mid 1 \leq i \leq t\}$, and $V_3^p := \{v_i^3 \mid 1 \leq i \leq t\}$, respectively. Assume that w is inserted at position $5 + 2t$ in more than $q/3$ votes of V_1^p. Then, d_1, which is not fixed in every vote of V_1^p, would beat c. Analogously, if w was inserted at position $5 + 2t$ in more than $q/3$ votes of V_2^p or V_3^p, then c would be beaten by h_1 or h_1', respectively. Now, we have that w must take position $5 + 2t$ in q votes and can take this position in at most $q/3$ votes from V_i^p, for every $i \in \{1, 2, 3\}$ and thus must take this position in exactly $q/3$ votes of V_1^p, V_2^p, and V_3^p.

Second, we show that the candidates from D_{12} ensure that w takes position $5 + 2t$ in v_i^1 if and only if w takes position $5+2t$ in v_i^2. The proof is by contradiction. Assume that there is an extension in which w takes position $5 + 2t$ in v_i^1 and another position in v_i^2 for any i. Since d_i and h_{i+1} have been shifted to the left in v_i^1, each of them can only be shifted to the left in at most $q/3 - 1$ further votes. By construction, v_i^2 is the only vote of $V_1^p \cup V_2^p$ in which neither d_i nor h_{i+1} is shifted to the left by setting w to position $5 + 2t$. However, since w can either take the first or position $2t + 5$ in an extension (as argued above), it must take the first position in v_i^2. Now, w has to take the position $5 + 2t$ in $2q/3 - 1$ further votes from $V_1^p \cup V_2^p$ and thus in each of these votes w will either shift d_i or h_{i+1}. Hence, either d_i or h_{i+1} must be shifted to the left in more than $q/3 - 1$ further votes and will beat c, a contradiction. The other case (w takes position $5 + 2t$ in v_i^2 and another position in v_i^1) follows in complete analogy by considering h_i and d_{i+1}. One can show analogously that the candidates of D_{13} ensure that w takes position $5 + 2t$ in v_i^1 if and only if it takes the same position in v_i^3. Thus,

Observation I follows.

Now, one has that Possible Winner is NP-hard for all scoring rules with a scoring vector of size $f((E,\mathcal{S}))$ with strictly decreasing score values. By using some simple padding, we extend the result for the remaining cases, that is for scoring vectors of size $m' > f((E,\mathcal{S}))$ and$f((E,\mathcal{S}))$ different score values. To this end, we introduce a set of $m' - f((E,\mathcal{S}))$ new dummy candidates and cast the linear votes such they cannot beat the distinguished candidate in any extension. The original candidates from C are placed on positions endued with strictly decreasing points, whereas the new candidates are placed on the remaining positions. Then, if the positions of candidates get shifted (when w is inserted), the "old" candidates are affected in the same manner as in the above construction and the theorem follows.

□

7.6 An unbounded number of positions with equal score values

In the previous section, we showed NP-hardness for scoring vectors with an unbounded number of different score values. In this section, we discuss scoring vectors with an unbounded number of positions with equal score value. In the first subsection, we show NP-hardness for Possible Winner for scoring vectors that fulfill $\alpha_2 \neq \alpha_{m-1}$, and, in the second subsection, we consider the special case that $\alpha_1 > \alpha_2 = \cdots = \alpha_{m-1} > 0$. Note that these two cases cover all scoring vectors with an unbounded number of equal score values (except plurality and veto): There are three ways to "violate" $\alpha_1 > \alpha_2 = \cdots = \alpha_{m-1} > 0$. First, if one requires $\alpha_1 = \alpha_2$, then one ends up with veto. Second, requiring $\alpha_{m-1} = 0$, one arrives at plurality. Third, requiring $\alpha_2 \neq \alpha_{m-1}$, then one ends up with the other case that includes the famous examples 2-approval and $(m-2)$-approval.

7.6.1 An unbounded number of positions with equal score values and $\alpha_2 \neq \alpha_{m-1}$

The scoring vectors considered in this subsection divide into two classes. First, there are at least two score values that are greater than the "equal score value". Second, there are at least two score values that are smaller than the "equal score value". Formally, a size-m scoring vector for the second class looks as follows: there is an i, with $i < m-2$ and an "unbounded" integer x such that $\alpha_{i-x} = \alpha_i > \alpha_{i+1}$. This property can be used to construct a basic "logical" tool used in the many-one reductions of this subsection: For two candidates c, c', having $c \succ c'$ in a partial vote implies that setting c such that it makes less than α_i points implies that also c' makes less than α_i points whereas all candidates placed in the range between $i-x$ and i make exactly α_i points. This can be used to model some implication of the type "$c \Rightarrow c'$" in a vote. For $(m-2)$-approval, which will play a prominent role for stating our results, this condition means that c only has the possibility to make zero points in a vote if also c' makes zero points in this vote whereas all other candidates make one point.

Most of the reductions of this subsection are from the NP-complete Multicolored Clique (MC) problem [105]:

Multicolored Clique (MC)
Given: An undirected graph $G = (X_1 \cup X_2 \cup \cdots \cup X_k, E)$ with $X_i \cap X_j = \emptyset$ for $1 \leq i < j \leq k$ and the vertices of X_i induce an independent set for $1 \leq i \leq k$.
Question: Is there a complete subgraph (clique) of size k?

Here, $1, \ldots, k$ are considered as different colors. Then, the problem is equivalent to ask for a *multicolored clique*, that is, a clique that contains one vertex for every color. To ease the presentation, for any $1 \leq i \neq j \leq k$, we interpret the vertices of X_i as red vertices and write $r \in X_i$, and the vertices of X_j as green vertices and write $g \in X_j$.

Reductions from MC are often used to show parameterized hardness results [105]. Intuitively, the different colors give some useful structure to the instance. The general idea is to construct different types of gadgets. Here, the partial votes realize four kinds of gadgets. First, gadgets that choose a vertex of every color (vertex selection). Second, gadgets that choose an edge of every ordered pair of colors, for example, one edge from green to red and one edge from red to green (edge selection). Third, gadgets that check the consistency of two selected ordered edges, e.g. does the chosen red-green candidate refer to the same edge as the choice of the green-red candidate (edge-edge match)? Finally, gadgets that check whether all edges starting from the same color start from the same vertex (vertex-edge match). Though reductions from MC have become a standard tool to obtain hardness results, the reduction given here is not straightforward. For example, we are not aware of any reduction in the literature for which it is necessary to employ vertex-edge match gadgets.

We start by giving a reduction from MC that settles the NP-hardness of Possible Winner for $(m-2)$-approval. Then we describe how the given construction can be generalized to work for most of the cases considered in this subsection. The NP-hardness of the remaining cases will be shown by reductions from Exact Cover By 3-Sets.

Lemma 7.2. Possible Winner *is NP-hard for* $(m-2)$*-approval.*

Proof. Given an MC-instance $G = (X, E)$ with $X = X_1 \cup X_2 \cup \cdots \cup X_k$. Let $E(i,j)$ denote all edges from E between X_i and X_j. Without loss of generality, we can assume that there are integers s and t such that $|X_i| = s$ for $1 \leq i \leq k$, $|E(i,j)| = t$ for all i, j, and that k is odd since every other instance can be padded easily in this way. We construct a partial profile P on a set C of candidates such that the distinguished candidate $c \in C$ is a possible winner if and only if there is a size-k clique in G. The set of candidates $C := \{c\} \uplus C_X \uplus C_E \uplus D$, where $\uplus$ denotes the disjoint union, is specified as follows:

- For $i \in \{1, \ldots, k\}$, let $C_X^i := \{r_1, \ldots, r_{k-1} \mid r \in X_i\}$ and $C_X := \bigcup_i C_X^i$.
- For $i, j \in \{1, \ldots, k\}, i \neq j$, let

$$C_{i,j} := \{rg \mid \{r, g\} \in E(i,j), r \in X_i, \text{ and } g \in X_j\}$$

and

$$C'_{i,j} := \{rg' \mid \{r, g\} \in E(i,j), r \in X_i, \text{ and } g \in X_j\}.$$

Then, $C_E := (\bigcup_{i \neq j} C_{i,j}) \uplus (\bigcup_{i \neq j} C'_{i,j})$, i.e., for every edge $\{r, g\} \in E(i,j)$, the set C_E contains the four candidates rg, rg', gr, gr'.

- The set $D := D_X \uplus D_1 \uplus D_2$ is defined as follows.
  For $i \in \{1, \ldots, k\}$, $D_X^i := \{c_1^r, \ldots, c_{k-2}^r \mid r \in X_i\}$ and $D_X := \bigcup_i D_X^i$.
  For $i \in \{1, \ldots, k\}$, one has $D_1^i := \{d_1^i, \ldots, d_{k-2}^i\}$ and $D_1 := \bigcup_i D_1^i$.
  The set D_2 is defined as $D_2 := \{d^i \mid i \in \{1, \ldots, k\}\}$.

We refer to the candidates of C_X as *vertex-candidates*, to the candidates of C_E as *edge-candidates*, and to the candidates of D as *dummy-candidates*.

The partial profile P consists of a set of linear votes V^l and a set of partial votes V^p. In each extension of P, the distinguished candidate c gets one point in every vote from V^p (see definition below). Thus, according to Lemma 7.1, we can set the maximum partial scores as follows. For every candidate $d^i \in D_2$, $s_p^{\max}(d^i) = |V^p| - s + 1$, that is, d^i must get zero points (*take a zero-position*) in at least $s - 1$ of the partial votes. For every remaining candidate $c' \in C \backslash (\{c\} \cup D_2)$, $s_p^{\max}(c') = |V^p| - 1$, that is, c' must get zero points in at least one of the partial votes.

In the following, we define $V^p := V_1 \cup V_2 \cup V_3 \cup V_4$. For all our gadgets only the last positions of the votes are relevant. Hence, in the partial votes it is sufficient to explicitly specify the "relevant candidates". More precisely, we define for all partial votes that each candidate that does not appear explicitly in the description of a partial vote is positioned before all candidates that appear in this vote.

The partial votes of V_1 realize the **edge selection gadgets**. Basically, selecting an ordered edge (r, g) with $\{r, g\} \in E$ means to select the corresponding *pair of edge-candidates* rg and rg'. The candidate rg is used for the vertex-edge match check and rg' for the edge-edge match check. Now, we give the definition of V_1. For every ordered color pair $(i, j), i \neq j$, V_1 has $t-1$ copies of the partial vote $\{rg \succ rg' \mid \{r, g\} \in E(i, j)\}$, that is, one partial vote contains the constraint $rg \succ rg'$ for every $\{r, g\} \in E(i, j)$. The idea of this gadget is as follows. For every ordered color pair we have t edges and $t-1$ corresponding votes. Within one vote, one pair of edge-candidates can get the two available zero-positions. Thus, it is possible to set all but one, namely the selected pair of edge-candidates, to zero-positions.

The partial votes of V_2 realize the **vertex selection gadgets**. Here, we will use the $k-1$ candidates corresponding to a selected vertex to do the vertex-edge match for all edges that are incident in a multicolored clique. Formally, we set $V_2 := V_2^a \cup V_2^b$ as further defined in the following. Intuitively, in V_2^a we select a vertex and in V_2^b, by a cascading effect, we achieve that all $k-1$ candidates that correspond to this vertex are selected. In V_2^a, for every color i, we have $s-1$ copies of the partial vote $\{r_1 \succ c_1^r \mid r \in X_i\}$. In V_2^b, for every color i and for every vertex $r \in X_i$, we have the following $k-2$ votes.

For all *odd* $z \in \{1, \ldots, k-4\}$, $\quad v_z^{r,i} : \{c_z^r \succ c_{z+1}^r, r_{z+1} \succ r_{z+2}\}$.
For all *even* $z \in \{2, \ldots, k-3\}$, $\quad v_z^{r,i} : \{c_z^r \succ c_{z+1}^r, d_{z-1}^i \succ d_z^i\}$,
$\quad v_{k-2}^{r,i} : \{c_{k-2}^r \succ d_{k-2}^i, r_{k-1} \succ d^i\}$.

The partial votes of V_3 realize the **vertex-edge match gadgets**. For $i, j \in \{1, \ldots, k\}$, for $j < i$, V_3 contains the vote $\{rg \succ r_j \mid \{r, g\} \in E, r \in X_i, \text{ and } g \in X_j\}$ and, for $j > i$, V_3 contains the vote $\{rg \succ r_{j-1} \mid \{r, g\} \in E, r \in X_i, \text{ and } g \in X_j\}$.

The partial votes of V_4 realize the **edge-edge match gadgets**. For every unordered color pair $\{i, j\}, i \neq j$ there is the partial vote $\{rg' \succ gr' \mid \{r, g\} \in E(i, j), r \in X_i, \text{ and } g \in X_j\}$.

This completes the description of the partial profile. Now, we verify a property of the construction that is crucial to see the correctness: In total, the number of zero-

positions available in the partial votes is exactly equal to the sum of the minimum number of zero-position the candidates of $C \backslash \{c\}$ must take such that c is a winner. We denote this property of the construction as *tightness*. To see the tightness property, we first compute the number of partial votes:

$$\begin{aligned} |V_1| + |V_2| + |V_3| + |V_4| = \\ k(k-1)(t-1) + k(s-1) + ks(k-2) + k(k-1) + k(k-1)/2 = \\ t(k^2-k) + s(k^2-k) + k^2/2 - 3k/2. \end{aligned} \tag{7.1}$$

Regarding the number of zero-positions that must be taken, we first compute the number of candidates for each subset:

- $|C_X| = sk(k-1)$,
- $|C_E| = 2tk(k-1)$,
- $|D_X| = sk(k-2)$, $|D_1| = k(k-2)$, and $|D_2| = k$.

The candidates of D_2 must take at least $s-1$ zero-positions and all other candidates at least one. Thus, in total the number of zero-positions must be at least

$$\begin{aligned} sk^2 - sk + 2tk^2 - 2tk + sk^2 - 2ks + k^2 - 2k + k(s-1) = \\ 2s(k^2-k) + 2t(k^2-k) + k^2 - 3k. \end{aligned} \tag{7.2}$$

Furthermore, there are two zero-positions for every partial vote. It is easy to verify that (7.1) times two equals (7.2). Hence, the tightness of the construction is shown. It directly follows that if there is a candidate that takes more zero-positions than desired, then c cannot win in this extension since then at least one zero-position must be "missing" for another candidate.

We can now show the following claim to complete the proof.

Claim: The graph G has a clique of size k if and only if c is a possible winner in P.

"$\Rightarrow$" Given a multicolored clique Q of G of size k. We refer to the vertices and edges belonging to Q as solution vertices and solution edges, respectively, and to the corresponding candidates as solution candidates. Then, extend the partial profile P as given in Figure 7.5. In the following we argue that in the given extension every candidate takes the required number of zero-positions.

In V_1, for every ordered color pair, all pairs of edge-candidates except the pair of solution edge-candidates are set to the last two positions in one of the $t-1$ votes.

In V_2^a for every color i, we set all candidates r_1 that do not belong to the solution vertices and the corresponding c_1^r to zero-positions in one of the votes. In V_2^b for every non-solution vertex $r \in X_i \backslash Q$ we set the corresponding candidates r_{z+1} and r_{z+2} at zero-positions in the votes $v_z^{r,i}$ with odd index $z \in \{1, \ldots, k-4\}$. In the votes with even index $z \in \{2, \ldots, k-3\}$, we set the corresponding dummy candidates c_z^r, c_{z+1}^r at zero-positions. We further set the candidate r_{k-1} at a zero-position in votes $v_z^{r,i}$ for all the $s-1$ non-solution vertices of color i, which implies that the dummy candidate d^i is placed at $s-1$ zero-positions. Thus, we have "enough" zero-positions for all the copies

V_1 :		$\cdots > rg > rg'$	for $i, j \in \{1, \ldots, k\}, i \neq j, r \in X_i \backslash Q$, and $g \in X_j \backslash Q$
V_2^a :		$\cdots > r_1 > c_1^r$	for $1 \leq i \leq k$ and $r \in X_i \backslash Q$
V_2^b :	$v_z^{r,i}$	$\cdots > r_{z+1} > r_{z+2}$	for $1 \leq i \leq k, r \in X_i \backslash Q$ for all $z \in \{1, 3, 5, \ldots, k-4\}$
	$v_z^{r,i}$	$\cdots > c_z^r > c_{z+1}^r$	for $1 \leq i \leq k, r \in X_i \backslash Q$ for all $z \in \{2, 4, 6, \ldots, k-3\}$
	$v_{k-2}^{r,i}$	$\cdots > r_{k-1} > d^i$	for $1 \leq i \leq k, r \in X_i \backslash Q$
	$v_z^{r,i}$	$\cdots > c_z^r > c_{z+1}^r$	for $1 \leq i \leq k, r \in X_i \cap Q$ for all $z \in \{1, 3, 5, \ldots, k-4\}$
	$v_z^{r,i}$	$\cdots > d_{z-1}^i > d_z^i$	for $1 \leq i \leq k, r \in X_i \cap Q$ for all $z \in \{2, 4, 6, \ldots, k-3\}$
	$v_{k-2}^{r,i}$	$\cdots > c_{k-2}^r > d_{k-2}^i$	for $1 \leq i \leq k, r \in X_i \cap Q$
V_3 :		$\cdots > rg > r_j$	for $i, j \in \{1, \ldots, k\}, j < i, r \in X_i \cap Q$, and $g \in X_j \cap Q$
		$\cdots > rg > r_{j-1}$	for $i, j \in \{1, \ldots, k\}, j > i, r \in X_i \cap Q$, and $g \in X_j \cap Q$
V_4 :		$\cdots > rg' > gr'$	for $i, j \in \{1, \ldots, k\}, i \neq j, r \in X_i \cap Q, g \in X_j \cap Q$

Figure 7.5: Extension of the partial votes for the MC-instance. Extensions in which candidates that do not correspond to the solution set Q take the zero-positions are highlighted.

of the non-solution candidates, the corresponding dummy candidates $\{c_1^r, \ldots, c_{k-2}^r \mid r \in X_i \setminus Q\}$, and d^i. The remaining votes of V_2^b "correspond" to the gadgets for the solution vertices. Here, we set the candidate pairs $c_z^r > c_{z+1}^r$ in the votes with odd index $z \in \{1, \ldots, k-4\}$ at position zero and the candidate pairs with candidates d_p^i for $p = 1, \ldots, k-2$ to zero-positions in the votes with even index. Thus, in V_2, we have improved c upon all dummy candidates and upon all candidates corresponding to non-solution vertices, whereas each candidate corresponding to a solution vertex must still take a zero-position.

Now, it remains to set every candidate corresponding to a solution vertex or a solution edge to a zero-position in at least one vote. Due to construction, for a solution edge $\{r, g\} \in E$, the two corresponding candidates rg' and gr' can be set to zero in the corresponding vote of V_4. And, in V_3 the $k-1$ vertex-candidates belonging to every solution vertex can be set to a zero-position in combination with the corresponding edge-candidate. Thus, the distinguished candidate c is the winner of the described extension.

"$\Leftarrow$" Given an extension of P in which c is a winner, we show that the "selected" candidates must correspond to a size-k clique. Recall that the number of zero-positions that each candidate must take is "tight" in the sense that if one candidate gets an unnecessary zero-position, then for another candidate there are not enough zero-positions left.

First (edge selection), for $i, j \in \{1, \ldots, k\}, i \neq j$, we consider the candidates of $C_{i,j}$. The candidates of $C_{i,j}$ can take zero-positions in one vote of V_3 and in $t-1$ votes of V_1. Since $|C_{i,j}| = t$ and in the considered votes at most one candidate of $C_{i,j}$ can take a zero-position, every candidate of $C_{i,j}$ must take one zero-position in one of these votes. We refer to a candidate that takes the zero-position in V_3 as solution candidate rg_{sol}. For every non-solution candidate $rg \in C_{i,j} \backslash \{rg_{\text{sol}}\}$, its placement in V_1 also implies that rg' gets a zero-position, whereas rg'_{sol} still needs to take one zero-position (which is only possible in V_4).

Second, we consider the vertex selection gadgets. Here, analogously to the edge selection, for every color i, we can argue that in V_2^a, out of the set $\{r_1 \mid r \in X_i\}$, we

have to set all but one candidate to a zero-position. The corresponding *solution vertex* is denoted as r_{sol}. For every vertex $r \in X_i \backslash \{r_{\text{sol}}\}$, this implies that the corresponding dummy-candidate c_1^r also takes a zero-position in V_2^a. Now, we show that in V_2^b we have to set all candidates that correspond to non-solution vertices to a zero-position whereas all candidates corresponding to r_{sol} must appear only at one-positions. Since for every vertex $r \in X_i \backslash \{r_{\text{sol}}\}$, the vertex c_1^r has already a zero-position in V_2^a, it cannot take a zero-position within V_2^b anymore without violating the tightness. In contrast, for the selected solution candidate r_{sol}, the corresponding candidates $c_1^{r_{\text{sol}}}$ and r_{sol_1} still need to take one zero-position. The only possibility for $c_1^{r_{\text{sol}}}$ to take a zero-position is within vote $v_1^{r_{\text{sol}},i}$ by setting $c_1^{r_{\text{sol}}}$ and $c_2^{r_{\text{sol}}}$ to the last two positions. Thus, one cannot set r_{sol_2} and r_{sol_3} to a zero-position within V_2. Hence, the only remaining possibility for r_{sol_2} and r_{sol_3} to get zero points remains within the corresponding votes in V_3. This implies for every non-solution vertex r that r_2 and r_3 cannot get zero points in V_3 and thus we have to choose to put them on zero-positions in the vote $v_1^{r,i}$ from V_2^b. The same principle leads to a cascading effect in the following votes of V_2^b: One cannot choose to set the candidates $c_p^{r_{\text{sol}}}$ for $p \in \{1, \ldots, k-2\}$ to zero positions in votes of V_2^b with even index z and thus has to improve upon them in the votes with odd index z. This implies that all vertex-candidates belonging to r_{sol} only appear in one-positions within V_2^b and that all dummy candidates d_p^i for $p \in \{1, \ldots, k-2\}$ are set to one zero-position. In contrast, for every non-solution vertex r, one has to set the candidates c_p^r, $p \in \{2, \ldots, k-2\}$, to zero-positions in the votes with even index z, and thus in the votes with odd index z, one has to set all vertex-candidates belonging to r to zero-positions. This further implies that for every non-solution vertex in the last vote of V_2^b one has to set d^i to a zero-position, and since there are exactly $s-1$ non-solution vertices, d^i takes the required number of zero positions. Altogether, all vertex-candidates belonging to a solution vertex still need to be placed at a zero-position in the remaining votes $V_3 \cup V_4$, whereas all dummy candidates of D and the candidates corresponding to the other vertices must have taken enough zero positions.

Third, consider the vertex-edge match realized in V_3. For $i, j \in \{1, \ldots, k\}, i \neq j$, there is only one remaining vote in which rg_{sol} with $r \in X_i$ and $g \in X_j$ can take a zero position. Hence, rg_{sol} must take this zero-position. This implies that the corresponding incident vertex-candidate x is also set to a zero-position in this vote. If $x \neq r_{sol_i}$, then x has already a zero-position in V_2. Hence, this would contradict the tightness and rg_{sol} and the corresponding vertex must "match". Furthermore, the construction ensures that each of the $k-1$ candidates corresponding to one vertex appears exactly in one vote of V_3 (for each of the $k-1$ candidates, the vote corresponds to edges from different colors). Hence, c can only be a possible winner if a selected vertex matches with all selected incident edges.

Finally, we discuss the edge-edge match gadgets. In V_4, for $i, j \in \{1, \ldots, k\}, i \neq j$, one still needs to set the solution candidates from $C_{i,j}$ to zero-positions. We show that this can only be done if the two "opposite" selected edge-candidates match each other. For two such edges rg_{sol} and gr_{sol}, $r \in X_i, g \in X_j$, there is only one vote in V_4 in which they can get a zero position. If rg_{sol} and gr_{sol} refer to different edges, then in this vote only one of them can get zero points, and thus the other one still beats c. Altogether, if c is a possible winner, then the selected vertices and edges correspond to a multicolored clique of size k. □

By generalizing the reduction used for Lemma 7.2, one can show the following.

Theorem 7.2. *An MC-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m scoring vector that fulfills the following. There is an $i \leq m-1$ such that $\alpha_{i-x} = \cdots = \alpha_{i-1} > \alpha_i$ with $x = f(I)$. A suitable poly-type function f can be computed in polynomial time.*

Proof. We describe how to modify the reduction given in the proof of Lemma 7.2 to work for the considered cases. For this, let P on C denote a partial profile as constructed in the proof of Lemma 7.2. Since $i \leq m-1$, the position $i+1$ must exist. We set $x = f(I) := |C| - 2$ and fill all positions smaller than $i - x$ and all positions greater than $i+1$ with dummy candidates that are different from candidates in C and that are beaten by c in every extension. We distinguish the two subcases $\alpha_i = \alpha_{i+1}$ (1a) and $\alpha_i \neq \alpha_{i+1}$ (1b).

For the case (1a), one can argue in complete analogy to Lemma 7.2 by "identifying" the two zero-positions of Lemma 7.2 with position i and $i+1$ and setting the maximum partial score as follows (which can be done without changing the partial votes due to Lemma 7.1). For all $d^i \in D_2$, $s_p^{\max}(d^i) = (s-1) \cdot \alpha_i + (|V^p| - s + 1) \cdot \alpha_{i-1}$ and for all $c' \in C \backslash (\{c\} \cup D_2)$, $s_p^{\max}(c') = \alpha_i + (|V^p| - 1) \cdot \alpha_{i-1}$.

For (1b), we need to argue that the tightness argument still holds. For this, we set the maximum partial scores as follows (which can be done without changing the partial votes due to Lemma 7.1). For all $d^i \in D_2$, $s_p^{\max}(d^i) = (s-1) \cdot \alpha_{i+1} + (|V^p| - s + 1) \cdot \alpha_{i-1}$ and, for all $c' \in C \backslash (\{c\} \cup D_2)$, $s_p^{\max}(c') = \alpha_i + (|V^p| - 1) \cdot \alpha_{i-1}$. Now, in any extension in which c wins, each candidate in D_2 must be placed at least $s-1$ times on position $i+1$, and each of the other candidates must be placed on position i or $i+1$ at least once. Then again, the number of positions i and $i+1$ that still have to be assigned to candidates is exactly equal to the number of candidates that need to take these positions, hence, the tightness argument still holds. Thus, the correctness of the modified reduction can be shown in complete analogy to Lemma 7.2. □

In the following, we consider scoring rules with an unbounded number x of equal positions for which it holds that there is an $i \geq 2$ such that $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$. Parts of the results are based on further extensions of the MC-reduction used to prove Lemma 7.2. After that there still remain some cases for which it seems even more complicated to adapt the MC-reduction. However, for these cases we can make use of other properties of the scoring rules and settle them by less involved reductions from Exact Cover by 3-Sets. As we will see in Section 7.7, the following Lemmata 7.3–7.6 cover all scoring vectors with $i \geq 2$ such that $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$.

Lemma 7.3. *An MC-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m scoring vector that fulfills the following. There is an $i \geq 2$ such that $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$ with $x = f(I)$* ***and*** *there is a position $j < i$ with $\alpha_j < 2\,\alpha_{j+1}$. A suitable poly-type function f can be computed in polynomial time.*

Proof. We describe how to modify the MC-reduction given in the proof of Lemma 7.2 to work for the considered case. For this, let P on C denote a partial profile as constructed in the proof of Lemma 7.2. First, we describe the construction for $j = i-1$, that is, one has $\alpha_{i-1} < 2\,\alpha_i$. We construct a partial profile $\widetilde{P}$ as follows. We set $x = f(I) = |C| - 2$ and all positions $< i-1$ and $> i+x$ are filled with dummy candidates that are beaten by c in every extension. The positions not filled with dummies "contain" the partial votes of P in "reverse" order: In P all relative orders

are given for pairs of candidates. In $\widetilde{P}$ we just "flip" every pair, for example, instead of having $rg \succ rg'$ we have $rg' \succ rg$ in V_1. We define that all candidates that are not given explicitly are worse than the given candidates in a vote (instead of being better). By flipping the order of a pair, we adapt the "logical implication", for example, instead of having "if rg makes zero points, then also rg' makes zero points" in P, we have "if rg makes α_i points, then also rg' makes at least α_i points" in $\widetilde{P}$. Furthermore, we set the maximum partial scores to $s_p^{\max}(d^i) = (s-1) \cdot \alpha_{i-1} + (|V^p| - s + 1) \cdot \alpha_{i+1}$ for all $d^i \in D_2$ and $s_p^{\max}(c') = \alpha_{i-1} + (|V^p| - 1) \cdot \alpha_{i+1}$ for all $c' \in C \backslash (\{c\} \cup D_2)$. Note that since $\alpha_{i-1} < 2\,\alpha_i$, every candidate c' can take either position i or position $i-1$ in one of the partial votes. Then, we can use a "reverse" tightness argument: Since the positions i and $i-1$ must be taken by two candidates in every vote and every candidate can take at most one such position (or at most $s-1$ such positions for candidates in D_2, respectively), by counting candidates and positions it holds that if every candidate of D_2 must make α_{i-1} points exactly $(s-1)$ times, then every other candidate must make α_{i-1} or α_i points exactly once. Thus, it remains to show that every $d^i \in D$ must take position $i-1$ in $s-1$ of the votes. Assume this is not the case, then there must be two votes $v_{k-2}^{r,i}$ and $v_{k-2}^{r',i}$ with $r \neq r'$ in which d^i does not take position $i-1$. Due to construction, the only remaining candidate that can take this position in these votes is d_{k-2}^i, but this is not possible due to $s_p^{\max}(d_{k-2}^i)$. Hence, we can use a tightness argument analogously to Lemma 7.2. Since we also adapted the logical implication, the correctness follows in complete analogy to Lemma 7.2.

The remaining cases $(j < i-1)$ follow by padding positions within the gadgets. More precisely, replace each specified pair, e.g. $rg' \succ rg$ by $rg' \succ rg \succ H$ with a dummy set H of size $i-(j+1)$ and replace α_{i-1} by α_j in the new definitions of the maximum partial scores. □

So far, we settled the NP-hardness for scoring vectors with $i \geq 2$ such that $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$ if there is a position $j < i$ with $\alpha_j < 2\,\alpha_{j+1}$. Without the constraint $\alpha_j < 2\,\alpha_{j+1}$, it seems pretty complicated to adapt the tightness property which is crucial for the MC-reduction. Fortunately, the remaining cases have some different properties that allow to settle them by less complicated reductions from Exact Cover By 3-Sets. More precisely, in the following, we give three reductions with increasing difficulty. (Although all three reductions are self-contained, they might be easier to understand when reading them in the given order.)

Lemma 7.4. *An X3C-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m scoring vector that fulfills the following. There is an $i \geq 2$ such that $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$ with $x = f(I)$* ***and*** *there is a position $j < i$ with $\alpha_j \geq 3\,\alpha_i$. A suitable poly-type function f for X3C can be computed in polynomial time.*

Proof. Let $(E, \mathcal{S})$ denote an X3C-instance. Construct a partial profile P on a set of candidates C. The set C of candidates is defined by $C := \{c\} \uplus S \uplus E \uplus H \uplus D$ where c denotes the distinguished candidate c, $S := \{s_z \mid S_z \in \mathcal{S}\}$, E the set of candidates that represent the elements of the universe, and H and D contain disjoint candidates such that the following hold. We define $H := \biguplus_{z=1}^{|\mathcal{S}|} H_z$ with $|H_z| = i - j$ for all $z \in \{1, \ldots, |\mathcal{S}|\}$ needed to "pad" some positions relevant to the construction and $|D| = m - |S| - |E| - |H| - 1$ needed to pad irrelevant positions. We refer to the candidates from S as *subset candidates* and to the candidates from E as *element*

candidates. Set $f((E,\mathcal{S})) := |C \setminus D| - (i-j)$. For $1 \leq z \leq |\mathcal{S}|$, let $S_z = \{e_{z1}, e_{z2}, e_{z3}\}$. The partial profile P consists of a set of linear votes and a set of partial votes V^p. In all votes of V^p, we pad all irrelevant positions, i.e. all positions smaller than j and greater than $j-1+|C \setminus D|$ by fixing candidates from D (omitted in the further description). The set V^p consists of $|\mathcal{S}| - |E|/3$ copies of the vote

$$s_1 \succ H_1 \succ C\backslash(S \cup H), s_2 \succ H_2 \succ C\backslash(S \cup H), \ldots, s_{|\mathcal{S}|} \succ H_{|\mathcal{S}|} \succ C\backslash(S \cup H)$$

denoted as V_1^p and the following three votes, denoted as $V_2^p(z)$, for every $s_z \in S$

$v_z^1 : \quad H_1 \succ \{s_z, e_{z1}\} \succ C\backslash(\{s_z, e_{z1}\} \cup H_1)$,
$v_z^2 : \quad H_1 \succ \{s_z, e_{z2}\} \succ C\backslash(\{s_z, e_{z2}\} \cup H_1)$,and
$v_z^3 : \quad H_1 \succ \{s_z, e_{z3}\} \succ C\backslash(\{s_z, e_{z3}\} \cup H_1)$.

The basic idea of this construction is that in V_1^p one has to set all but $|E|/3$ "subset" candidates to position j whereas the remaining candidates will be able to take a position greater than i in all votes from V_1^p. Therefore, the remaining $|E|/3$ subset candidates can make $\alpha_j - \alpha_{i+1}$ points more than the other candidates within the remaining votes. This will enable them to shift their corresponding element candidates to position $i+1$ by taking position i. Since $\alpha_j > 3 \cdot \alpha_i$, they will be able to shift all three element candidates, respectively. To realize the basic idea, we adapt the maximum partial scores appropriately. For $e \in E$, let n_e denote the number of subsets in $\mathcal{S}$ which contain e. Then according to Lemma 7.1, we can cast the linear votes such that the following holds:

- $s_p^{\max}(s_z) = \alpha_j + (|V^p| - 1) \cdot \alpha_{i+1}$, for all $s_z \in S$,
- $s_p^{\max}(e) = (n_e - 1) \cdot \alpha_i + (|V^p| - n_e + 1) \cdot \alpha_{i+1}$, for all $e \in E$, and
- all other candidates are beaten by c in every extension.

We show that c is a possible winner in P if and only if there is an exact 3-cover for $(E, \mathcal{S})$:

Assume there is an exact 3-cover Q. Then one extends P by setting each s_z with $S_z \notin Q$ at position j in one vote from V_1^p and the corresponding candidates from H_z to the positions $j+1, \ldots, i$ in the same vote. Furthermore, set s_z to position $i+1$ in v_z^1, v_z^2, and v_z^3. Now, we have that every s_z with $S_z \notin Q$ takes position j in one vote and a position greater than i in all remaining votes and thus is beaten by c. This also means that in V_1^p all positions $\leq i$ are filled and thus every candidate s_z with $S_z \in Q$ takes a position greater than i in all votes from V_1^p. Thus, the remaining votes can be extended by setting every s_z with $S_z \in Q$ to position i in v_z^1, v_z^2, and v_z^3. Since $\alpha_j \geq 3\,\alpha_i$, the maximum partial score of s_z is not exceeded. Because Q is an exact 3-cover, all element candidates are shifted to position $i+1$ in one vote and thus are beaten by c. Hence, c is a winner in the described extension.

For the other direction, consider an extension of P in which c wins. Due to construction, in V_1^p only subset candidates from S can take position j. Because of the maximum partial scores, position j must be taken by different candidates from S in the $|\mathcal{S}| - |E|/3$ votes of V_p^1. We denote these candidates as non-solution candidates and the remaining $|E|/3$ candidates from S as solution candidates. Due to $s_p^{\max}(s_z)$, every non-solution candidate must take position $i+1$ in all remaining votes and thus the corresponding element candidates must make α_i points in the corresponding votes.

Hence, there remain only $|E|/3$ solution candidates that have to "shift" the $|E|$ element candidates to position $i+1$. Since every solution candidate can shift at most 3 candidates, the solution candidates must correspond to an exact 3-cover. □

In the following lemma, we consider a more specific type of scoring vector in the sense that there are only two score values greater than zero. This restriction allows us to find an easy way to "lift" the condition "$\alpha_j \geq 3 \cdot \alpha_i$" for two special types of scoring rules that will be sufficient for the proof of the main result in Section 7.7. Compared to the reduction from the previous lemma, for the following cases we also choose a set of "solution subset candidates" within the first part of the partial votes, but we will need some additional gadgetry to be able to "shift" the corresponding element candidates.

Lemma 7.5. *An X3C-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m scoring vector $(\alpha_1, \alpha_2, 0, \ldots, 0)$ with $3\,\alpha_2 > \alpha_1 > 2\,\alpha_2$ and $m = f(I) + 2$. A suitable poly-type function f can be computed in polynomial time.*

Proof. Let $(E, \mathcal{S})$ denote an X3C-instance. Construct a partial profile P on a set of candidates C as follows. The set of candidates consists of a distinguished candidate c, a set $S := \{s_i \mid S_i \in \mathcal{S}\}$ (the subset candidates), a set $D := \{d_i \mid S_i \in \mathcal{S}\}$, the set E (the element candidates), a candidate x, and $H := \{h_1, \ldots, h_{|\mathcal{S}|}\}$. Set $f((E, \mathcal{S})) := |C| - 2$. For $1 \leq i \leq |\mathcal{S}|$, let $S_i = \{e_{i1}, e_{i2}, e_{i3}\}$. The partial profile P consists of a set of linear votes and a set of partial votes V^p. The set V^p consists of $|\mathcal{S}| - |E|/3$ copies of the vote

$$s_1 \succ h_1 \succ C \backslash (S \cup H), s_2 \succ h_2 \succ C \backslash (S \cup H), \ldots, s_{|\mathcal{S}|} \succ h_{|\mathcal{S}|} \succ C \backslash (S \cup H)$$

denoted as V_1^p and the following three votes for every $S_i \in \mathcal{S}$

$v_i^1 : \quad d_i \succ e_{i1} \succ C \backslash \{d_i, e_{i1}, s_i\}, s_i \succ C \backslash \{d_i, e_{i1}, s_i\}$
$v_i^2 : \quad x \succ \{d_i, e_{i2}\} \succ C \backslash \{d_i, e_{i2}, x\}$
$v_i^3 : \quad x \succ \{d_i, e_{i3}\} \succ C \backslash \{d_i, e_{i3}, x\}$

Let n_e denote the number of subsets in which e occurs. Then, due to Lemma 7.1, we can set the maximum partial scores as follows:

- $s_p^{\max}(s_i) = \alpha_1$ for all $s_i \in S$,
- $s_p^{\max}(d_i) = 3 \cdot \alpha_2$ for all $d_i \in D$,
- $s_p^{\max}(e) = (n_e - 1) \cdot \alpha_2$ for all $e \in E$,
- all other candidates are beaten by c in every extension.

We show that c is a possible winner in P if and only if there is an exact 3-cover for $(E, \mathcal{S})$:

Assume there is an exact 3-cover Q for $(E, \mathcal{S})$. Then we extend P as follows. For every $S_i \notin Q$, s_i takes position 1 and h_i takes position 2 in one vote from V_1^p and s_i takes position 3 in v_i^1. The corresponding d_i takes position 3 in v_i^2 and v_i^3. Clearly, for $S_i \notin Q$, $s_p^{\max}(s_i)$ is not exceeded, $s_p(d_i) = \alpha_1 < 3\,\alpha_2 = s_p^{\max}(d_i)$, and within V_1^p all first positions are fixed. For every solution set $S_i \in Q$, we set s_i to a position greater than 2 in all votes from V_1^p and to the first position in v_i^1. Since this

implies that d_i takes the second position in v_i^1, this enables us to set d_i to the second position in v_i^2 and v_i^3 without violating $s_p^{\max}(d_i)$. Since Q is an exact 3-cover, all corresponding element candidates are shifted to the third position once and for every element candidate the maximum partial score is not exceeded. Hence, c is a winner.

To see the other direction, assume there is an extension in which c wins. In V_1^p, the first positions can only be taken by candidates from S. Since each $s_i \in S$ can get α_1 points exactly once, $|\mathcal{S}|-|E|/3$ different subset candidates from S have to be placed on the first position. Let the set consisting of these candidates be denoted by S'. Every candidate s_i from S' has exploited its maximum partial score and therefore has to be placed on the third position in v_i^1. This implies that the corresponding candidate d_i takes the first position in v_i^1. Since $\alpha_1 > 2\,\alpha_2$ and $s_p^{\max}(d_i) = 3\,\alpha_2$, d_i has to take the third position in both v_i^2 and v_i^3. Hence, for $s_i \in S'$, the corresponding element candidates e_{i1}, e_{i2}, e_{i3} receive α_2 points each. However, each of the element candidates from E has to be placed on position 3 at least once due to its maximum partial score. This can only be in the remaining partial votes, that is, all v_i^1, v_i^2, v_i^3 with $s_i \in S \setminus S'$. Since $|S \setminus S'| = |E|/3$, one must shift one element candidate in each of these votes. For this, the only possibility is to set every $s_i \in S \setminus S'$ to position 1 in v_i^1, and the corresponding candidate d_i takes the second position in v_i^2 and v_i^3. Since c wins, all $|E|$ element candidates must get shifted to position 3. Hence, $S \setminus S'$ corresponds to an exact 3-cover of $(E, \mathcal{S})$. □

Finally, we settle the NP-hardness for a specific scoring vector.

Lemma 7.6. *An X3C-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m scoring vector $(2, 1, 0, \ldots, 0)$ for $m = f(I)+2$. A suitable poly-type function f can be computed in polynomial time.*

Proof. Let $(E, \mathcal{S})$ denote an X3C-instance. Construct a partial profile P on a set of candidates C as follows. The set of candidates consists of a distinguished candidate c, a set $S := \{s_i \mid S_i \in \mathcal{S}\}$ (the subset candidates), $D := \{d_i \mid S_i \in \mathcal{S}\}$, $T := \{t_i \mid S_i \in \mathcal{S}\}$, E (the element candidates), a candidate y, and $X := \{x_1, \ldots, x_{|\mathcal{S}|-|E|/3}\}$. Set $f((E, \mathcal{S})) := |C| - 2$. For $1 \le i \le |\mathcal{S}|$, let $S_i = \{e_{i1}, e_{i2}, e_{i3}\}$. The partial profile P consists of a set of linear votes and a set of partial votes V^p. The set $V^p := V_1^p \cup V_2^p \cup V_3^p$ is further defined as follows. The set V_1^p consists of $|\mathcal{S}| - |E|/3$ copies of the partial vote

$$s_1 \succ t_1 \succ C\backslash(S \cup T), s_2 \succ t_2 \succ C\backslash(S \cup T), \ldots, s_{|\mathcal{S}|} \succ t_{|\mathcal{S}|} \succ C\backslash(S \cup T).$$

The set V_2^p consists of $|\mathcal{S}| - |E|/3$ copies of the partial vote

$$y \succ T \succ C\backslash(T \cup \{y\})$$

and V_3^p contains the following three votes for every $S_i \in \mathcal{S}$

$$\begin{array}{ll} v_i^1: & d_i \succ e_{i1} \succ C\backslash\{d_i, e_{i1}, s_i\}, s_i \succ C\backslash\{d_i, e_{i1}, s_i\} \\ v_i^2: & y \succ \{d_i, e_{i2}\} \succ C\backslash\{d_i, e_{i2}, y\} \\ v_i^3: & \{t_i, e_{i3}\} \succ C\backslash(\{t_i, e_{i3}\} \cup X) \end{array}$$

Let n_e denote the number of subsets in which e occurs and $n_{e,3}$ the number of subsets in which e is denoted as e_{i3} for $i \in \{1, \ldots, |S|\}$. Then, using Lemma 7.1, we set the maximum partial scores as follows:

V_1^p :		$s_i > t_i > \dots$	for $S_i \notin Q$
V_2^p :		$y > t_i > \dots$	for $S_i \notin Q$
V_3^p :	v_i^1	$d_i > e_{i1} > s_i > \dots$	for $S_i \notin Q$
	v_i^2	$y > e_{i2} > d_i > \dots$	for $S_i \notin Q$
	v_i^3	$e_{i3} > x_q > t_i \dots$	for $S_i \notin Q$ and different q
	v_i^1	$s_i > d_i > e_{i1} > \dots$	for $S_i \in Q$
	v_i^2	$y > d_i > e_{i2} > \dots$	for $S_i \in Q$
	v_i^3	$t_i > e_{i3} > \dots$	for $S_i \in Q$

Figure 7.6: Extension for the X3C-reduction for the case $(2, 1, 0, \dots)$. The remark "different q" means that for $i \neq i'$ with $S_i \notin Q$ and $S_{i'} \notin Q$ one chooses two different candidates from X. Extensions corresponding to non-solution candidates are highlighted.

- $s_p^{\max}(s_i) = s_p^{\max}(t_i) = s_p^{\max}(d_i) = 2$ for $i \in \{1, \dots, |\mathcal{S}|\}$
- $s_p^{\max}(x_i) = 1$ for $i \in \{1, \dots, |\mathcal{S}| - |E|/3\}$
- $s_p^{\max}(e) = 2n_{e,3} + (n_e - n_{e,3}) - 1$ for $e \in E$
- the candidate y is beaten by c in every extension

We show that c is a possible winner in P if and only if there is an exact 3-cover for $(E, \mathcal{S})$:

Assume there is an exact 3-cover Q for $(E, \mathcal{S})$. Then we extend P as given in Figure 7.6. For every $S_i \notin Q$, s_i takes the first position in one vote from V_1^p and makes zero points in all remaining votes. The corresponding t_i takes the second position in one vote from V_1^p and one vote from V_2^p and makes zero points in all remaining votes. Hence, c beats these s_i and t_i and the votes from V_1^p and V_2^p are fixed. For every $S_i \notin Q$, we extend v_i^3 by setting a different candidate from X at the second position such that none of them is put on this position twice, and hence c also beats every candidate from X. For every $S_i \in Q$, d_i, t_i and s_i make exactly 2 points in V_3^p and thus are beaten by c as well. It remains to consider the element candidates. To this end, note that a candidate $e \in E$ is beaten by c if there is an i such that e takes position 3 in v_i^1 or v_i^2 or takes position 2 in v_i^3. Since Q is an exact 3-cover and all candidates corresponding to subsets from Q are shifted to the right in one vote, c wins in the given extension.

To see the other direction, assume there is an extension in which c wins. Let $G^1 := \{v_i^1 \mid 1 \leq i \leq |\mathcal{S}|\}$, $G^2 := \{v_i^2 \mid 1 \leq i \leq |\mathcal{S}|\}$, and $G^3 := \{v_i^3 \mid 1 \leq i \leq |\mathcal{S}|\}$. We start by arguing that at most $2/3 \cdot |E|$ candidates from E can make zero points in a vote from $G^1 \cup G^2$. For any i, at most two element candidates, namely e_{i1} and e_{i2} can make zero points in $G^1 \cup G^2$. More precisely, due to $s_p^{\max}(d_i)$, if s_i takes the first position in v_i^1, then e_{i1} and e_{i2} can take the third position and if s_i takes the second position, then only e_{i1} can be shifted to the third position, since d_i takes the first position in v_i^1 and has exploited its maximum partial score. Thus, the number of points that all candidates from S can make within V_3^p is an upper bound for the number of element candidates that can be shifted. Since only candidates from S can take the first positions in V_1^p, $|V_1^p| = |\mathcal{S}| - |E|/3$, and $s_p^{\max}(s_i) = 2$, the candidates from S can make at most $2/3|E|$ points in V_3^p. Thus, there are at most $2/3|E|$ element candidates

that can take a position with zero points in $G^1 \cup G^2$. Thus, due to $s_p^{\max}(e)$, in G^3 one must shift (at least) $|E|/3$ candidates to the second position (**Observation 1**). In the following, we show that the only way to do so leads to an extension in which exactly $|E|/3$ candidates s_i from S make zero points in V_1^p and the corresponding t_i make zero points in $V_1^p \cup V_2^p$ whereas all other candidates from $S \cup T$ have already accomplished their maximum partial score in $V_1^p \cup V_2^p$ (**Claim 1**). This means that the element candidates that are shifted to the right correspond to exactly $|E|/3$ subsets $S_i \in \mathcal{S}$. Since every element candidate must be shifted at least once, these subsets must form an exact 3-cover in $(E, \mathcal{S})$.

We use a tightness criterion (analogously to the MC-reduction from Lemma 7.2) to prove Claim 1. To this end, we show that the score of all positions that must be filled equals the sum of the maximum partial scores of all candidates. Again, it directly follows that a candidate $c' \in C \backslash \{c\}$ cannot make less than $s_p^{\max}(c')$ points since otherwise there must be another candidate that beats c. Now, we show the tightness. The total number of votes is

$$|V_1^p| + |V_2^p| + |V_3^p| = |\mathcal{S}| - |E|/3 + |\mathcal{S}| - |E|/3 + 3|\mathcal{S}| = 5|\mathcal{S}| - 2/3|E|.$$

In V_2^p and V_3^p, candidate y is already fixed at the first position in $2|\mathcal{S}| - 1/3|E|$ votes and since in every vote 3 points have to be given, there are $3 \cdot (5|\mathcal{S}| - 2/3|E|) - 2 \cdot (2|\mathcal{S}| - 1/3|E|) = 11|\mathcal{S}| - 4/3|E|$ points for the remaining candidates left. The sum of the maximum partial scores from all candidates from $S \cup T \cup D \cup X \cup E$ is

$$3 \cdot 2 \cdot |\mathcal{S}| + |\mathcal{S}| - |E|/3 + 2|\mathcal{S}| + 2|\mathcal{S}| - |E| = 11|\mathcal{S}| - 4/3|E|.$$

To see this, note that clearly $\sum_{e \in E} n_{e,3} = |\mathcal{S}|$ and $\sum_{e \in E} n_e = 3|\mathcal{S}|$. Thus, the tightness follows.

Now, we finally show the correctness of Claim 1. Due to the tightness, the $|\mathcal{S}| - |E|/3$ candidates from X must take position 2 in $|\mathcal{S}| - |E|/3$ votes from G^3. Thus, there remain $|E|/3$ second positions in G^3 that are not fixed. Note that due to tightness, a candidate e_{i3} cannot take the third position in v_i^3. Hence, if the remaining second positions are not taken by candidates from E, we shift less than $|E|/3$ candidates in G^3, a contradiction to Observation 1. Hence, these positions must be taken by candidates from E and thus all second positions within G^3 are fixed. This implies that every candidate t_i from T must take either the first or the third position in v_i^3. More precisely, since $|E|/3$ candidates from E take a second position there must be $|E|/3$ candidates from T that take the first positions within the corresponding votes. However, a candidate from T can only take the first position if it makes zero points in $V_1^p \cup V_2^p$. Hence, there must be $|E|/3$ candidates from T, denoted as T', that make zero points in $V_1^p \cup V_2^p$ and, due to tightness, all remaining candidates from T must make 2 points in $V_1^p \cup V_2^p$. A candidate $t_i \in T$ can make at most one point in V_1^p since due to the condition "$s_i \succ t_i$" it shifts s_i to the first position (and $s_p^{\max}(s_i) = 2$). Hence, making two points within $V_1^p \cup V_2^p$ implies that t_i must make one point in V_1^p and one point in V_2^p and that the corresponding s_i must make 2 points in V_1^p. This fixes all positions in $V_1^p \cup V_2^p$ and since a candidate s_i with $t_i \in T'$ clearly makes zero points in $V_1^p \cup V_2^p$, the correctness of Claim 1 follows. Altogether, we have that $\{S_i \mid t_i \in T'\}$ forms an exact 3-cover for $(E, \mathcal{S})$. □

7.6.2 Scoring vectors with $\alpha_1 > \alpha_2 = \cdots = \alpha_{m-1} > 0$

In this subsection, we consider scoring rules defined by scoring vectors that fulfill $\alpha_1 > \alpha_2 = \cdots = \alpha_{m-1} > 0$. Although quite special, these rules might be of interest of their own. They can be considered as a direct combination of the very common plurality and veto rules where one allows to weight the contribution of the plurality or veto part. For example, by using $(10, 1, \ldots, 1, 0)$ the "plurality" part would have more influence to the outcome, whereas for $(10, 9, \ldots, 9, 0)$ the "veto" part would be more important. To show NP-hardness, we give two types of many-one reductions from X3C; one for the case $\alpha_1 < 2 \cdot \alpha_2$ and one for the case $\alpha_1 > 2 \cdot \alpha_2$. As mentioned before, the case $\alpha_1 = 2 \cdot \alpha_2$ remains open. Intuitively, for all other cases we make use of the "asymmetry" of the differences of the score values, that is, by shifting a candidate from the first to the second position one decreases its score by a different amount than by shifting it from the last but one to the last position. In the two following proofs, the position in a linear order in which a candidate gets α_1 points is denoted as *top position*, a position in which a candidate gets α_2 points as *middle position*, and the position in which a candidate gets zero points as *last position*.

Theorem 7.3. *An X3C-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m scoring vector satisfying the conditions $\alpha_1 > \alpha_2 = \alpha_{m-1} > \alpha_m = 0$ and $\alpha_1 < 2 \cdot \alpha_2$ for $m = f(I) + 2$. A suitable poly-type function f can be computed in polynomial time.*

Proof. Let $(E, \mathcal{S})$ denote an X3C-instance. We construct a partial profile P for which the distinguished candidate $c \in C$ is a possible winner if and only if $(E, \mathcal{S})$ is a yes-instance. The set of candidates is $C := \{c, h\} \uplus \{s_i, d_i, t_i \mid S_i \in \mathcal{S}\} \uplus E$. The partial profile P consists of a set of partial votes V^p and a set of linear orders V^l. For $1 \leq i \leq |\mathcal{S}|$, let $S_i = \{e_{i1}, e_{i2}, e_{i3}\}$. Then the set of partial votes $V^p := V_1^p \cup V_2^p$ is given by the following subsets. The set V_1^p consists of $|E|/3$ copies of the partial vote

$$h \succ C \setminus \{h, s_1, \ldots, s_{|\mathcal{S}|}\} \succ \{s_1, \ldots, s_{|\mathcal{S}|}\}.$$

For every $i \in \{1, \ldots, |\mathcal{S}|\}$, the set V_2^p contains the three votes
$v_i^1 : h \succ C \setminus \{h, s_i, d_i\} \succ \{s_i, d_i\}$,
$v_i^2 : e_{i1} \succ C \setminus \{e_{i1}, t_i, d_i\} \succ t_i$, and
$v_i^3 : e_{i2} \succ C \setminus \{e_{i2}, e_{i3}, t_i\} \succ e_{i3}$.

Now, we pass on to the definitions of the maximum partial scores. To this end, for a candidate e corresponding to an element $e \in E$ (referred to as element candidate), let $n_{e,1+2}$ denote the number of subsets from $\mathcal{S}$ in which e is identical with e_{i1} or e_{i2}. Due to Lemma 7.1, we can cast the linear votes such that the following hold:

- $s_p^{\max}(s_i) = (|V^p| - 1) \cdot \alpha_2$,
- $s_p^{\max}(d_i) = s_p^{\max}(t_i) = \alpha_1 + (|V^p| - 2) \cdot \alpha_2$,
- $s_p^{\max}(e) = (|V^p| - n_{e,1+2} + 1) \cdot \alpha_2 + (n_{e,1+2} - 1) \cdot \alpha_1$,
- h is beaten by c in every extension.

The maximum partial scores of the element candidates are set such that every element candidate has to be "shifted" to the right at least once. More precisely, if a

V_1^p :		$h >$	$\dots$	$> s_i$	$S_i \in Q$
V_2^p :	v_i^1	$h >$	$\dots$	$> s_i > d_i$	$S_i \in Q$
	v_i^2	$d_i > e_{i1} >$	$\dots$	$> t_i$	$S_i \in Q$
	v_i^3	$t_i > e_{i2} >$	$\dots$	$> e_{i3}$	$S_i \in Q$
	v_i^1	$h >$	$\dots$	$> d_i > s_i$	$S_i \notin Q$
	v_i^2	$e_{i1} >$	$\dots$	$> t_i > d_i$	$S_i \notin Q$
	v_i^3	$e_{i2} >$	$\dots$	$> e_{i3} > t_i$	$S_i \notin Q$

Figure 7.7: Extension for the case $\alpha_1 > \alpha_2 = \alpha_{m-1} > 0$ and $\alpha_1 < 2 \cdot \alpha_2$. Extensions for candidates that do not correspond to subsets belonging to the solution set Q are highlighted.

candidate e took the first position in all votes in which it is identical with e_{i1} or e_{i2} and the second position in all remaining votes (including the votes in which it is identical with e_{i3}), then $s(e) = (|V^p| - n_{e,1+2}) \cdot \alpha_2 + n_{e,1+2} \cdot \alpha_1 > s_p^{\max}(e)$ since $\alpha_1 > \alpha_2$. However, if, for any i, t_i or d_i are inserted at the first position in one of the votes in which e appears, then e makes at least $\alpha_1 - \alpha_2$ points less and thus is beaten by c. We denote this as **Observation 2**. Now, we show the correctness of the construction.

Claim: Candidate c is a possible winner in P if and only if $(E, \mathcal{S})$ is a yes-instance.

"$\Leftarrow$": Let Q denote an exact 3-cover for $(E, \mathcal{S})$. Then extend P as displayed in Figure 7.7. More precisely, within V_1^p every candidate s_i with $S_i \in Q$ takes the last position in exactly one of the $|E|/3$ votes. Then, the candidates make the following points within the extension of the partial votes. Every s_i takes the last position in one vote and middle positions in all other votes and thus makes exactly $s_p^{\max}(s_i)$ points. For $S_i \in Q$, every candidate t_i and every candidate d_i takes one first and one last position, and thus, $s(d_i) = s(t_i) = \alpha_1 + (|V^p| - 2) \cdot \alpha_2 = s_p^{\max}(d_i) = s_p^{\max}(t_i)$. In the corresponding votes every element candidate is shifted once since Q is an exact 3-cover and thus is beaten by c due to Observation 2. Clearly, for $S_i \notin Q$, s_i is beaten by c as well. It remains to consider d_i and t_i with $S_i \notin Q$. Here, one has $s(d_i) = (|V^p| - 1) \cdot \alpha_2 < s_p^{\max}(d_i)$ and $s(t_i) = (|V^p| - 1) \cdot \alpha_2 < s_p^{\max}(t_i)$. Hence, c beats all other candidates and wins.

"$\Rightarrow$": Consider an extension in which c wins. Due to $s_p^{\max}(s_i)$, every candidate s_i must take the last position in at least one of the votes. Since $|V_1^p| = |E|/3$, at most $|E|/3$ candidates can take a last position in V_1^p; denote the set of them by S'. Hence at least $|\mathcal{S}| - |E|/3$ candidates s_i must take the last position in v_i^1. Now, we show that for these candidates the corresponding element candidates cannot be shifted to the right in v_i^2 or v_i^3. Since s_i takes the last position in v_i^1, d_i already makes $(|V^p| - 1) \cdot \alpha_2$ in the extended partial votes without v_i^2. Hence, d_i must take the last position in v_i^2 since otherwise $s(d_i) = |V^p| \cdot \alpha_2 > s_p^{\max}(d_i)$ because $\alpha_1 < 2\,\alpha_2$. This implies that e_{i1} is not shifted and that t_i takes a middle position in v_i^2. Now, for t_i it follows analogously that t_i must take the last position in v_i^3 and thus neither e_{i2} nor e_{i3} is shifted. Altogether, this means that all element candidates must be shifted by candidates from S'. Every $s_i \in S'$ can shift three candidates by setting s_i in the last position in v_i^1 and d_i and t_i to the first positions in v_i^2 and v_i^3, respectively. Since there are $|E|$ element candidates, it follows that $|S'| = |E|/3$ and that all $s_i \in S'$ must shift disjoint sets of element candidates. Hence, S' corresponds to an exact 3-cover

for $(E, \mathcal{S})$. □

In the remainder of this subsection, we consider the case that $\alpha_1 > 2 \cdot \alpha_2$. We also give a reduction from X3C. Note that the previous proof cannot be transferred directly and thus we give a modified construction for which it will be more laborious to show the correctness.

Theorem 7.4. *An X3C-instance I can be reduced to a* Possible Winner*-instance for a scoring rule which produces a size-m-scoring vector satisfying the conditions $\alpha_1 > \alpha_2 = \alpha_{m-1} > \alpha_m = 0$ and $\alpha_1 > 2 \cdot \alpha_2$ for $m = f(I) + 2$. A suitable poly-type function f can be computed in polynomial time.*

Proof. Let $(E, \mathcal{S})$ denote an X3C-instance. Let k denote the size of a solution for $(E, \mathcal{S})$, that is, $k := |E|/3$, and $t := |\mathcal{S}|$. We construct a partial profile P for which the distinguished candidate $c \in C$ is a possible winner if and only if $(E, \mathcal{S})$ is a yes-instance. The set of candidates is $C := S \uplus D \uplus E \uplus \{c, h\}$ with $S := \{s_i \mid 1 \leq i \leq t\}$ (the subset candidates) and $D := \{d_i \mid 1 \leq i \leq t\}$, and E (the element candidates).

Very roughly, the basic idea of the reduction is as follows. There are three subsets of partial votes, in the first subset V_1^p one "selects" $t - k$ subset candidates from S that do not correspond to an exact 3-cover and in the second subset V_2^p one selects k subset candidates that correspond to an exact 3-cover. Selecting hereby means that a solution subset candidate gets zero points in one vote of V_2^p whereas every non-solution candidate gets α_1 points in a vote of V_1^p. Hence, a solution candidate can make more points than a non-solution candidate in the third subset V_3^p. Thus, a solution candidate can take a top position in V_3^p which yields a cascading effect that makes it possible to shift the corresponding element candidates such that they do not beat the distinguished candidate c.

Formally, the partial profile P consists of a set of partial votes V^p and a set of linear orders V^l. For $1 \leq i \leq t$, let $S_i = \{e_{i1}, e_{i2}, e_{i3}\}$, then the set of partial votes $V^p := V_1^p \cup V_2^p \cup V_3^p$ is given by the following subsets.

V_1^p:	$t - k$ copies of the partial vote		$S \succ C \backslash (S \cup \{h\}) \succ h$
V_2^p:	k copies of the partial vote		$h \succ C \backslash (S \cup \{h\}) \succ S$
V_3^p:	for $1 \leq i \leq t$ the three partial votes	w_1^i:	$d_i \succ C \backslash \{d_i, e_{i1}, s_i\} \succ e_{i1}$
		w_2^i:	$h \succ C \backslash \{d_i, e_{i2}, h\} \succ \{e_{i2}, d_i\}$
		w_3^i :	$h \succ C \backslash \{d_i, e_{i3}, h\} \succ \{e_{i3}, d_i\}$

Note that in w_1^i, candidate s_i can be inserted at any position. The distinguished candidate c makes α_2 points in every partial vote from V^p. Hence, according to Lemma 7.1, we can set the linear orders of V^l such that the following holds. For $i = 1, \ldots, t$,

$$s_p^{\max}(s_i) = (|V^p| - 2) \cdot \alpha_2 + \alpha_1,$$

$$s_p^{\max}(d_i) = (|V^p| - 2) \cdot \alpha_2 + \alpha_1 - z$$

with $z = \alpha_1 \mod \alpha_2$ if $\alpha_1 < 3\,\alpha_2$, and $z = \alpha_2$, otherwise[2]. Note that it holds

[2]Note that this maximum partial score does not exactly fulfill the conditions of Lemma 7.1 if $z \neq \alpha_2$. However, the construction can be easily extended to work for this case as well. More precisely, in this case $z = \alpha_1 - \lfloor \alpha_1 / \alpha_2 \rfloor \cdot \alpha_2$ and $\lfloor \alpha_1 / \alpha_2 \rfloor \leq 3$. Thus, in the construction given in the proof of Lemma 7.1 one can add α_1 and "subtract" α_2 as often as required. The subtraction can be accomplished by changing the role of the dummy "d" and d_i within a block.

V_1^p :		$s_i > C\backslash\{s_i, h\} > h$	$\forall s_i$ with $S_i \notin \mathcal{S}'$
V_2^p :		$h > C\backslash\{s_i, h\} > s_i$	$\forall s_i$ with $S_i \in \mathcal{S}'$
V_3^p :	w_1^i	$d_i > C\backslash\{s_i, d_i\} > s_i$	$\forall s_i$ with $S_i \notin \mathcal{S}'$
	w_2^i	$h > C\backslash\{d_i, h\} > d_i$	$\forall s_i$ with $S_i \notin \mathcal{S}'$
	w_3^i	$h > C\backslash\{d_i, h\} > d_i$	$\forall s_i$ with $S_i \notin \mathcal{S}'$
	w_1^i	$s_i > C\backslash\{s_i, e_{i1}\} > e_{i1}$	$\forall s_i$ with $S_i \in \mathcal{S}'$
	w_2^i	$h > C\backslash\{e_{i2}, h\} > e_{i2}$	$\forall s_i$ with $S_i \in \mathcal{S}'$
	w_3^i	$h > C\backslash\{e_{i3}, h\} > e_{i3}$	$\forall s_i$ with $S_i \in \mathcal{S}'$

Figure 7.8: Extension of V^p for an exact 3-cover $\mathcal{S}' \subseteq \mathcal{S}$. The middle positions are not given explicitly since the order of the candidates is irrelevant. Extensions for candidates which do not belong to the solution set $\mathcal{S}'$ are highlighted.

that $\alpha_2 \geq z$ and

$$\alpha_1 - z \geq 2\,\alpha_2\,. \tag{7.3}$$

For all $e \in E$, $s_p^{\max}(e) = (|V^p| - 1) \cdot \alpha_2$, that is, e must have the last position in one of the partial votes. And, $s_p^{\max}(h) \geq |V^p| \cdot \alpha_1$, that is, h can beat c in no extension.

We now prove the following claim.

Claim: Candidate c is a possible winner of (V, C) if and only if $(E, \mathcal{S})$ is a yes-instance for X3C.

"$\Leftarrow$": Let $S' \subseteq \mathcal{S}$ denote an exact 3-cover for $(E, \mathcal{S})$. Then, we extend the partial profile as follows (Figure 7.8). If $S_i \in S'$, then s_i is placed at the last position in one vote of V_2^p and at a middle position in all other votes from $V_1^p \cup V_2^p$. If $S_i \notin S'$, then s_i is placed at the first position in one of the votes in V_1^p and at a middle position in all other votes from $V_1^p \cup V_2^p$. This is possible since there are $t-k$ top position and k last positions that can be taken by candidates from S in $V_1^p \cup V_2^p$. In V_3^p, every candidate s_i with $S_i \in \mathcal{S}'$ is placed at the top position and the corresponding element candidates e_{i2}, e_{i3} at the last position in the respective votes. Every candidate s_i with $S_i \notin \mathcal{S}'$ is placed at the last position and the corresponding element candidates e_{i2}, e_{i3} are placed at a middle position.

In the described extension, the candidates make the following points in V^p. Every candidate $s_i \in S$ takes exactly one top position and exactly one last position in V^p. Hence $s(s_i) = s_p^{\max}(s_i)$. For the candidates of D one has to distinguish two cases. First, if $S_i \notin S$, then, $s(d_i) = (|V^p| - 3) \cdot \alpha_2 + \alpha_1 \leq s_p^{\max}(d_i)$ since $\alpha_2 \geq z$. Second, if $S_i \in S$, then $s(d_i) = |V^p| \cdot \alpha_2 = (|V^p| - 2) \cdot \alpha_2 + 2\,\alpha_2 \leq (|V^p| - 2) \cdot \alpha_2 + \alpha_1 - z = s_p^{\max}(d_i)$ because of Inequality (7.3). Finally, we have to consider the candidates from E. Since for every S_i in the 3-cover, the corresponding element candidates e_{i1}, e_{i2}, and e_{i3} get at the last position, every candidate of E takes one last and $|V^p| - 1$ middle positions and thus makes $(|V^p| - 1) \cdot \alpha_2$ points. It follows that c wins in the considered extension.

"$\Rightarrow$": In an extension of V in which c is the winner, every element candidate from E must take the last position in one vote of V^p. This is only possible in V_3^p since every element candidate is already fixed at a middle position in $V_1^p \cup V_2^p$. More precisely, for every i, e_{i1} gets a last position if s_i is inserted at a middle or the top position in the corresponding vote w_1^i and e_{i2}/e_{i3} can get a last position only if d_i takes a middle position in the corresponding vote w_2^i/w_3^i.

To find out what this means for the other candidates, we have to go into details here. For $i = 1, \ldots, t$, let b_i denote the "benefit", i.e., the maximum number of element candidates that can be put at a last position in V_3^p depending on where s_i is placed in w_1^i. Then, we can show the following.

Observation 3:

1. $b_i = 3$ if s_i is placed in a top position in w_1^i.
2. $b_i = 1$ if s_i is placed in a middle position in w_1^i.
3. $b_i = 0$ if s_i is placed in a last position in w_1^i.

To see Observation 3, note that if s_i is on the top position in w_1^i, then d_i can take the middle position in w_2^i or w_3^i since the corresponding score $s(d_i) = |V^p| \cdot \alpha_2 \leq s_p^{\max}(d_i)$. Thus, all three element candidates can be shifted to the last position. If s_i is not placed on the top, but in the middle position, then e_{i1} is still shifted to the last position, but d_i must take the last position in w_2^i or w_3^i and thus neither e_{i2} nor e_{i3} can have a last position in w_2^i or w_3^i. To see this, assume that d_i has the top position in w_1^i and a middle position in w_2^i or w_3^i, then

$$s(d_i) \geq |V_1^p \cup V_2^p| \cdot \alpha_2 + (|V_3^p| - 2) \cdot \alpha_2 + \alpha_1 = (|V^p| - 2) \cdot \alpha_2 + \alpha_1 > s_p^{\max}(d_i),$$

a contradiction. If s_i is placed on the last position in w_1^i, then e_{i1} cannot take the last position in V_3^p, and neither can e_{i2} and e_{i3}, because d_i takes the first position in w_1^i and gets α_1 points and has to take the last position in both w_2^i and w_3^i by the same argument as before.

In the following, we show that in an extension in which c wins, in V_1^p there must be $t - k$ different subset candidates s_i that take the top position and each of the remaining k (solution) candidates of S must take one last position in V_2^p. It directly follows by Observation 3 that for all non-solution candidates we must have that $b_i = 0$ and thus every solution candidate must shift the three corresponding element candidates that must be different from the element candidates corresponding to the other solution candidates.

For every i, let t_i denote the number of top positions that s_i takes within V_1^p and l_i the number of last positions that s_i takes within V_2^p. Observe that the following conditions must hold.

$$\begin{aligned}
&\sum_{i=1}^{t} l_i = k, \\
&\sum_{i=1}^{t} t_i = t - k, \text{ since every position must be taken,} \\
&\sum_{i=1}^{t} b_i \geq 3k, \text{ since there are } 3k \text{ element candidates and each} \\
&\qquad\qquad \text{one must take at least one last position.}
\end{aligned} \tag{7.4}$$

In the following, our strategy consists of three steps:

- We first investigate the dependencies of l_i, t_i, and b_i upon each other. For that sake, we distinguish the cases $l_i = 0$, $l_i = 1$, and $l_i \geq 2$.

- Second, based on these case distinctions, we can show that the case $l_i \geq 2$ is not possible, that is, every s_i can have at most one last position in V_2^i. This will need the most technical effort and will directly imply $t_i \leq 1$ for all i.
- Third, we show that there is no candidate s_i with $l_i = t_i = 1$, which will imply that only candidates with $l_i = 1$ contribute with a positive benefit and can place their element candidates at a last position. Since there are only k such candidates, they must correspond to an exact 3-cover.

First step. We show some dependencies of l_i, t_i, and b_i by systematically enumerating all possible cases. (In the argumentation that follows the case distinction we are only interested in upper bounds of b_i. Hence, we omit to show lower bounds.)

Case I: $l_i = 0$ **a)** if $t_i = 0$, then $b_i \leq 1$,
b) if $t_i = 1$, then $b_i = 0$,
c) $t_i \geq 2$ is not possible.

Proof of Case I:
Ia) ($l_i = t_i = 0$): Assume $b_i = 3$, i.e., s_i is on the top position in w_1^i due to Observation 3. Then $s(s_i) = (|V^p| - 1)\,\alpha_2 + \alpha_1 > s_p^{\max}(s_i)$, a contradiction, hence $b_i \leq 1$.
Ib) ($l_i = 0$, $t_i = 1$): Assume $b_i = 1$, i.e., s_i is on a middle position in w_1^i due to Observation 3. Then $s(s_i) = (|V^p| - 1)\,\alpha_2 + \alpha_1 > s_p^{\max}(s_i)$, a contradiction, hence $b_i = 0$.
Ic) ($l_i = 0$, $t_i \geq 2$): Assume s_i takes the last position in w_1^i, that is, s_i makes as few points as possible within this case. Then,

$$\begin{aligned} s(s_i) &= (|V^p| - t_i - 1)\,\alpha_2 + t_i\,\alpha_1 \\ &> (|V^p| - t_i - 1 + 2(t_i - 1))\,\alpha_2 + \alpha_1 \\ &> s_p^{\max}(s_i), \end{aligned}$$

a contradiction, hence this case is not possible.

Case II: $l_i = 1$ **a)** if $t_i = 0$, then $b_i \leq 3$,
b) if $t_i = 1$, then $b_i \leq 1$,
c) $t_i \geq 2$ is not possible.

Proof of Case II:
IIa) ($l_i = 1$, $t_i = 0$), trivial upper bound.
IIb) ($l_i = t_i = 1$) Assume $b_i = 3$, i.e., s_i is on the top position in w_1^i due to Observation 3. Then $s(s_i) = (|V^p| - 3)\,\alpha_2 + 2\,\alpha_1 > s_p^{\max}(s_i)$, a contradiction, hence $b_i \leq 1$.
IIc) ($l_i = 1$, $t_i \geq 2$): Even if s_i takes the last position in w_1^i one has

$$\begin{aligned} s(s_i) &= (|V^p| - t_i - 2)\,\alpha_2 + t_i\,\alpha_1 \\ &> (|V^p| - t_i - 2 + 2(t_i - 1))\,\alpha_2 + \alpha_1 \\ &= (|V^p| + t_i - 4)\,\alpha_2 + \alpha_1 \\ &\geq s_p^{\max}(s_i), \end{aligned}$$

a contradiction, hence this case is not possible.

Case III: $l_i \geq 2$ **a)** if $t_i = l_i$, then $b_i = 0$,
b) if $t_i = l_i - 1$, then $b_i \leq 1$,
c) if $t_i \leq l_i - 2$, then $b_i \leq 3$,
d) $t_i > l_i$ is not possible.

Proof of Case III:
IIIa) ($l_i \geq 2$, $t_i = l_i$): Assume $b_i = 1$, i.e., s_i is on a middle position in w_1^i due to Observation 3. Then

$$\begin{aligned} s(s_i) &= (|V^p| - t_i - l_i)\,\alpha_2 + t_i\,\alpha_1 \\ &= (|V^p| - 2t_i)\,\alpha_2 + t_i\,\alpha_1 \\ &> (|V^p| - 2t_i + 2(t_i - 1))\,\alpha_2 + \alpha_1 \\ &= (|V^p| - 2)\,\alpha_2 + \alpha_1 \\ &= s_p^{\max}(s_i), \end{aligned}$$

a contradiction, hence $b_i = 0$.
IIIb) ($l_i \geq 2, t_i = l_i - 1$): Assume $b_i = 3$, i.e., s_i is on the top position in w_1^i due to Observation 3, then

$$\begin{aligned} s(s_i) &= (|V^p| - t_i - l_i - 1)\,\alpha_2 + (t_i + 1)\,\alpha_1 \\ &= (|V^p| - 2t_i - 2)\,\alpha_2 + (t_i + 1)\,\alpha_1 \\ &> (|V^p| - 2t_i - 2 + 2t_i)\,\alpha_2 + \alpha_1 \\ &= (|V^p| - 2)\,\alpha_2 + \alpha_1 \\ &= s_p^{\max}(s_i), \end{aligned}$$

a contradiction, hence $b_i \leq 1$.
IIIc) ($l_i \geq 2$, $t_i \leq l_i - 2$): trivial upper bound.
IIId) ($l_i \geq 2, t_i > l_i$): Then

$$\begin{aligned} s(s_i) &= (|V^p| - t_i - l_i - 1)\,\alpha_2 + t_i\,\alpha_1 \\ &> (|V^p| - t_i - l_i - 1 + 2(t_i - 1))\,\alpha_2 + \alpha_1 \\ &= (|V^p| + t_i - l_i - 3)\,\alpha_2 + \alpha_1 \\ &\geq s_p^{\max}(s_i), \end{aligned}$$

a contradiction, hence this case is not possible.

Second step. Using the previous case distinctions, we show that no subset candidate s_i can take more than one last position in V_2^p. For this, without loss of generality, we assume that the candidates s_i are sorted in decreasing order according to their corresponding l_i, i.e.,

$$\underbrace{s_1, \ldots, s_j}_{l_i \geq 2}, \underbrace{s_{j+1}, \ldots, s_r}_{l_i = 1}, \underbrace{s_{r+1}, \ldots, s_t}_{l_i = 0}.$$

Claim 1: In an extension in which c wins, it holds that $l_i \leq 1$ for all i.

To prove Claim 1, we show that $j = 0$. More specifically, we prove that $j > 0$ implies that the total benefit $B := \sum_{i=1}^{t} b_i$ is less than $3k$. This means that not all $3k$ element candidates can take a last position and thus c cannot win.

Assume that $j > 0$. We start to show how to distribute the last and the first positions of V_1^p and V_2^p in order to maximize B. For that sake, let $T_j := \sum_{i=1}^{j} t_i$ denote the number of top positions that were taken by the first j candidates $s_1, \ldots, s_j$. Now, we consider the remaining indices $i \in \{j+1, \ldots, t\}$. Since for all of them $l_i \leq 1$,

it must also hold $t_i \leq 1$ (see Case I and Case II). Thus and because of Equation (7.4), there must be at least $t - k - T_j$ candidates from $s_{j+1}, \ldots, s_t$ with $t_i = 1$. For both remaining cases ($l_i = 1$ and $l_i = 0$), the benefit b_i is greater for the case $t_i = 0$ than it is for the case $t_i = 1$ (cf. Case I and Case II). Hence, to maximize the total benefit B, it is desirable to minimize the number of candidates having $t_i = 1$. Since there are $t-j$ indices greater than j and t_i must be equal to one for at least $t-k-T_j$ indices, there are at most $t - j - (t - k - T_j) = k + T_j - j$ indices with $t_i = 0$ (**Observation 4**). Furthermore, for every index from $\{j+1, \ldots, s_r\}$, by setting t_i to zero or one, one can "choose" between $b_i = 1$ and $b_i = 3$ (Case II). For the remaining indices, one can choose between $b_i = 0$ and $b_i = 1$ by setting t_i to zero or one (Case I). We show by contradiction that choosing Case IIa (which results in $b_i = 3$) as often as possible is the way to maximize B:

Assume that Case IIa holds, that is $l_i = 1$ and $t_i = 0$, is not chosen as often as possible. Then, first, there must be an index $i \in \{j+1, \ldots, r\}$ with $t_i = 1$ and hence with $b_i = 1$ (Case IIb). Second, there must be an index $x > r$ with $t_x = 0$ and hence $b_x = 1$ (Case Ia). Then setting $t_i = 1$ and $t_x = 0$ does not violate Equation (7.4) and has the following effect.

- b_i is increased by 2 (from 1 to 3),
- b_x is decreased by 1 (from 1 to 0).

Thus, $B = \sum_{i=1}^{t} b_i$ was not maximal.

Now, we have argued that to maximize B, one has to choose Case IIa as often as possible (**Observation 5**). Using this, we can compute the maximal value $\max B$ of B (showing that is must be less than $3k$). For that sake, we first consider the benefit coming from the first j candidates $s_1, \ldots, s_j$, which we denote by $B_j := \sum_{i=1}^{j} b_i$. Let B_j^0 denote the set of indices $i \in \{1, \ldots, j\}$ with $b_i = 0$, let B_j^1 denote the set of indices $i \in \{1, \ldots, j\}$ with $b_i = 1$, and let B_j^3 denote the set of indices $i \in \{1, \ldots, j\}$ with $b_i = 3$. Then, Case III directly gives the following bound for the number of top positions assumed by the first j candidates.

$$T_j \leq \sum_{i \in B_j^0} l_i + \sum_{i \in B_j^1} (l_i - 1) + \sum_{i \in B_j^3} (l_i - 2) = \sum_{i=1}^{j} l_i - |B_j^1| - 2|B_j^3|, \tag{7.5}$$

which will be needed in the following.

Due to the previous discussion we know that in the remaining positions, we have to choose $t_i = 0$ for $k + T_j - j$ indices (cf. Observation 4) and one should choose Case IIa, that is, $l_i = 1$ and $t_i = 0$, as often as possible (cf. Observation 5). Clearly, $l_i = 1$ must be chosen $k - \sum_{i=1}^{j} l_i$ times whereas there are $k + T_j - j$ indices with $t_i = 0$. Hence, to compute a total upper bound on B, we have to distinguish two cases: First, $k - \sum_{i=1}^{j} l_i \leq k + T_j - j$, and, second, $k - \sum_{i=1}^{j} l_i > k + T_j - j$.

For the first case, we obtain

$$\begin{aligned}
\max B &= \underbrace{|B_j^1| + 3|B_j^3|}_{B_j} + 3\underbrace{(k - \sum_{i=1}^{j} l_i)}_{l_i=1,\, t_i=0} + \underbrace{k + T_j - j - (k - \sum_{i=1}^{j} l_i)}_{l_i=0,\, t_i=0} \\
&= |B_j^1| + 3|B_j^3| + 3k - 2 \cdot \sum_{i=1}^{j} l_i + T_j - j \\
&\overset{(7.5)}{\leq} |B_j^1| + 3|B_j^3| + 3k - 2 \cdot \sum_{i=1}^{j} l_i + \sum_{i=1}^{j} l_i - |B_j^1| - 2|B_j^3| - j \\
&= 3k - \sum_{i=1}^{j} l_i - j + |B_j^3|
\end{aligned}$$

Since $|B_j^3| \leq j$ it holds that the maximal value of B is strictly less than $3k$ for $j \geq 1$. Thus, at least one element candidate does not take a last position and hence beats c, a contradiction.

For the second case, we obtain

$$\begin{aligned}
\max B &= \underbrace{|B_j^1| + 3|B_j^3|}_{B_j} + 3\underbrace{(k + T_j - j)}_{l_i=1,\, t_i=0} + \underbrace{k - \sum_{i=1}^{j} l_i - (k + T_j - j)}_{l_i=1,\, t_i=1} \\
&= |B_j^1| + 3|B_j^3| + 3k + 2T_j - 2j - \sum_{i=1}^{j} l_i \\
&\overset{(7.5)}{\leq} |B_j^1| + 3|B_j^3| + 3k + \sum_{i=1}^{j} l_i - |B_j^1| - 2|B_j^3| + T_j - 2j - \sum_{i=1}^{j} l_i \\
&= 3k + |B_j^3| + T_j - 2j
\end{aligned}$$

Furthermore, in this case it follows directly from $k - \sum_{i=1}^{j} l_i > k + T_j - j$ that $\sum_{i=1}^{j} l_i + T_j < j$. For $j > 0$ this means that $T_j < j$. By definition, we have $|B_j^3| \leq j$, and thus $\max B$ is less than $3k$. This completes the proof of Claim 1. We therefore have $j = 0$ which means $l_i \leq 1$ for all $i \in \{1, \ldots, t\}$ and thus also $t_i \leq 1$ for all i (Case I and II).

Third step. We now show that there cannot be any candidate s_i which takes one last position and one first position in $V_1 \cup V_2$, i.e. we cannot have $t_i = l_i = 1$ for any s_i. Assume that the set of candidates $Q := \{s_i \mid t_i = l_i = 1\}$ is not empty. Then, due to Observation 3, the maximum value of B is

$$\max B = \underbrace{1 \cdot |Q|}_{l_i=t_i=1} + \underbrace{3 \cdot (k - |Q|)}_{l_i=1, t_i=0} + \underbrace{0}_{l_i=0, t_i=1} + \underbrace{1 \cdot |Q|}_{l_i=t_i=0} = 3k - |Q|,$$

a contradiction. Thus, $t-k$ many of the subset candidates s_i take a top position in V_1^p, and the remaining k subset candidates take a last position in V_2^p. Now, each of these k candidates must place its corresponding element candidates at the last positions in V_3^p. Since c can only be a winner if each of the $3k$ element candidates takes a last position

in a vote from V_3^p and in total at most $3k$ element candidates can take a last position in V_3^p, every element candidate must take exactly one last position. Thus, for $i \neq j$ such that s_i and s_j take a last position in V_2^p, $\{e_{i1}, e_{i2}, e_{i3}\}$ and $\{e_{j1}, e_{j2}, e_{j3}\}$ must be disjoint. It follows that $\{S_i \mid s_i \text{ takes a last position in } V_2^p\}$ forms an exact 3-cover. □

7.7 Putting all together

We are now ready to combine the many-one reductions from the previous sections to one general reduction. Basically, the problem we encounter by using one specific reduction from the previous sections is that such a reduction produces a Possible Winner-instance with a certain number m of candidates. Thus, one needs to ensure that the size-m scoring vector provides a sufficient number of positions with equal/different scores. This seems not to be possible in general. However, for every specific instance of Exact Cover By 3-Sets or Multicolored Clique, we can compute a number of positions with equal or different scores that is sufficient for the corresponding reduction, and we can use the maximum of all these numbers for the combined reduction. This is the underlying idea for the following proof.

Theorem 7.5. Possible Winner *is NP-complete for a scoring rule r if there is a constant z such that all scoring vectors produced by r for more than z candidates are different from* $(0, \ldots, 0)$, $(1, 0, \ldots, 0)$, $(1, \ldots, 1, 0)$, *and* $(2, 1, \ldots, 1, 0)$.

Proof. We give a reduction from X3C restricted to instances of size greater than z to Possible Winner for r. Let I with $|I| > z$ denote an X3C-instance. Since X3C and MC are NP-complete, there is a polynomial-time reduction from X3C to MC. Hence, let I' denote an MC-instance whose size is polynomial in $|I|$ and which is a yes-instance if and only if I is a yes-instance.

Let f_1 denote a poly-type function to compute the number of different score values as stated for Theorem 7.1, f_1' as for Theorem 7.2, f_2' as for Lemma 7.3, f_2 as for Lemma 7.4, f_3 as for Lemma 7.5, f_4 as for Lemma 7.6, and f_5 as for Theorem 7.4. Define $x := \max\{f_1(I),\ f_1'(I'),\ f_2'(I'),\ f_2(I),\ f_3(I),\ f_4(I),\ f_5(I)\}$ and consider the scoring vector $\overrightarrow{\alpha}$ of size $x \cdot (x+1)$ produced by r. Then we show the following.

Claim: For $\overrightarrow{\alpha}$ it holds that $|\{i \mid \alpha_i > \alpha_{i+1}\}| \geq x$ or that $\alpha_i = \cdots = \alpha_{i+x}$ for some position i.

The correctness of the claim can be seen as follows. First, assume that $\overrightarrow{\alpha}$ does not fulfill $\alpha_i > \alpha_j$ for x different positions i. Then consider $x \cdot (x+1)$ indices of $\overrightarrow{\alpha}$. Since they can have at most x different score values, there must be a single score value that is assigned to at least $x+1$ indices, that is, there is an index i with $\alpha_i = \cdots = \alpha_{i+x}$. Second, if there is no index i such that $\alpha_i = \cdots = \alpha_{i+x}$ for a position i, then again consider $x \cdot (x+1)$ indices of $\overrightarrow{\alpha}$. Since each score value can be assumed at most x times, there must be at least x different score values.

Now, due to the Claim, we can distinguish two main cases. If $\overrightarrow{\alpha}$ has at least x different score values, then we apply the X3C-reduction given in Theorem 7.1. Otherwise, we have an unbounded number of equal score values. In this case we distinguish the subcases given in Table 7.3. For all these subcases, there are many-one reductions used in the corresponding lemmata/theorems. Hence, it remains to show that each scoring vector can be handled by at least one of these cases. Clearly, $\overrightarrow{\alpha}$ must have the form $\alpha_{i-x} = \cdots = \alpha_{i-1} > \alpha_i$ for an $i \leq m-1$ (Case I), or $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$

Table 7.3: Subcases for scoring rules having an unbounded number of equal score values.

Case I	$\exists i \leq m-1$ s.t.	$\alpha_{i-x} = \cdots = \alpha_{i-1} > \alpha_i$	Theorem 7.2
Case IIa	$\exists i \geq 2$, $\exists j < i$ s.t.	$\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$ and $\alpha_j < 2\,\alpha_{j+1}$	Lemma 7.3
Case IIb	$\exists i \geq 2$, $\exists j < i$ s.t.	$\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$ and $\alpha_j \geq 3\,\alpha_i$	Lemma 7.4
Case IIc		$(\alpha_1, \alpha_2, 0, \ldots, 0)$ and $3\,\alpha_2 > \alpha_1 > 2\,\alpha_2$	Lemma 7.5
Case IId		$(2, 1, 0, \ldots, 0)$	Lemma 7.6
Case III		$\alpha_1 > \alpha_2 = \alpha_{m-1} > \alpha_m = 0$ and $\alpha_1 \neq 2 \cdot \alpha_2$	Theorem 7.4

for $i \geq 2$ (Case II), or $\alpha_1 > \alpha_2 = \alpha_{m-1} > \alpha_m = 0$ and $\alpha_1 \neq 2 \cdot \alpha_2$ (Case III). For Case I and Case III, the existence of many-one reductions follows immediately from the corresponding Theorems 7.2 and 7.4. Thus, it remains to discuss Case II, the case that $\overrightarrow{\alpha}$ has the form $\alpha_i > \alpha_{i+1} = \cdots = \alpha_{i+x}$ for $i \geq 2$.

To this end, we start with the case $i > 2$. Clearly, there must be at least three scoring values which are not equal to zero, namely, $\alpha_{i-2}, \alpha_{i-1}$, and α_i. If one has $\alpha_{i-1} < 2\,\alpha_i$ or $\alpha_{i-2} < 2\,\alpha_{i-1}$, then NP-hardness follows directly from Lemma 7.3. Otherwise, one must have $\alpha_{i-1} \geq 2\,\alpha_i$ and $\alpha_{i-2} \geq 2\,\alpha_{i-1}$. Hence, it follows that $\alpha_{i-2} \geq 4\,\alpha_i$ and NP-hardness follows directly from Lemma 7.4. It remains to consider all scoring rules of type $(\alpha_1, \alpha_2, 0, \ldots, 0)$. Here, we can distinguish the following four cases:

- $\alpha_1 < 2\,\alpha_2$: NP-hardness follows from Lemma 7.3,
- $\alpha_1 = 2\,\alpha_2$: NP-hardness follows from Lemma 7.6,
- $2\,\alpha_2 < \alpha_1 < 3\,\alpha_2$: NP-hardness follows from Lemma 7.5, and
- $\alpha_1 \geq 3\,\alpha_2$: NP-hardness follows from Lemma 7.4.

Since the membership in NP is obvious, the main theorem follows. □

Pure scoring rules. Based on all previous considerations and [14, Theorem 2], we arrive at a full dichotomy for pure scoring rules. More precisely, we can state the following.

Theorem 7.6. Possible Winner *is solvable in polynomial time for plurality and veto and NP-complete for all other non-trivial pure scoring rules.*

Proof. Plurality and veto are polynomial-time solvable due to Proposition 7.1. Having any non-trivial scoring vector different from $(1, 0, \ldots, 0)$, $(1, \ldots, 1, 0)$, and $(2, 1, \ldots, 1, 0)$ for m candidates, it is not possible to obtain a scoring vector of one of these three types (or $(0, \ldots, 0)$) for $m' > m$ by inserting scoring values. Hence, since we only consider pure scoring rules, the scoring rule does not produce a scoring vector of type plurality, veto, $(0, \ldots, 0)$, or $(2, 1, \ldots, 1, 0)$ for all $m \geq z$. Then the statement follows by Theorem 7.5 and [14, Theorem 2]. □

"Non-pure" scoring rules. We end this section with a brief informal discussion about the problem of classifying non-pure scoring rules in general. As stated in Theorem 7.5, we can show NP-hardness for non-pure scoring rules if (starting from a constant) all produced scoring vectors are "difficult". Clearly, it is possible to extend

the range of NP-hardness results to scoring rules that produce only few "easy" vectors; for example, having a difficult vector for all odd numbers of candidates and an easy vector for all even ones. However, this is not possible in general. Roughly speaking, if the underlying difficult part of the language becomes too sparse, then there cannot be a many-one reduction from an NP-complete problem since the densities of the problems are not polynomially related (see e.g. [176]). Note that this situation does not appear for the dichotomy result from Hemaspaandra and Hemaspaandra [129] for Manipulation for weighted voters. The intuitive reason for this is that their reductions for the NP-hardness in the case of weighted voters already hold for a constant number of candidates (and all scoring rules except plurality are NP-hard in this case).

7.8 Conclusion

We settled the computational complexity for Possible Winner for almost all pure scoring rules. More precisely, the only case that was left open regards the scoring rule defined by the scoring vector $(2, 1, \ldots, 1, 0)$, whereas for all other rules except plurality and veto, we obtained NP-completeness results. In follow-up work, Baumeister and Rothe [14] completed the dichotomy by showing the NP-completeness of Possible Winner for the case of $(2, 1, \ldots, 1, 0)$.

Dichotomy results are particularly desirable since the provide a full classification. Our result also provides an easy-to-check condition to distinguish between hard and easy cases for a whole class of problems. In our case, somewhat surprisingly, the Possible Winner problems turned out to be NP-complete even for very simple scoring rules like 2-approval. This motivates a study of further parameterizations as provided in the following two chapters. More specifically, Chapter 8 is concerned with several single parameterizations for several scoring rules whereas in Chapter 9 we obtain fixed-parameter tractability with respect to combined parameterizations for Possible Winner under the k-approval protocol.

Chapter 8

A parameterized complexity study for scoring rules

The Possible Winner problem asks whether some distinguished candidate may become the winner of an election when the given incomplete votes are extended into complete ones in a favorable way. As discussed in the previous chapter, Possible Winner is NP-complete for all pure scoring rules except plurality and veto. This motivates a broader algorithmic study of Possible Winner by pursuing a multivariate complexity analysis to identify tractable scenarios. We investigate how different parameterizations influence the computational complexity of Possible Winner for scoring rules. For some parameterizations we obtain results holding for all scoring rules whereas for other parameterizations, we focus on specific scoring rules like Borda and k-approval. Besides investigating the computational complexity for the standard voting parameterizations "number of candidates" and "number of votes", we introduce problem-specific parameterizations measuring the amount of incompleteness. For definitions regarding Possible Winner, we refer to Section 7.1. In the following, we briefly discuss related work and give an overview of our results for the considered parameterizations.

Number of candidates. Since the number of candidates is often small compared to the number of votes, for example, in political elections studying how a bounded number of candidates influences the computational complexity is an important task. Walsh [193] showed that Possible Winner can be solved in polynomial time when the number of candidates is constant. However, the given algorithm does not imply fixed-parameter tractability with respect to the "number of candidates". We show that for all scoring rules, Possible Winner is fixed-parameter tractable with respect to the "number of candidates". The result is achieved by using integer linear programming in combination with a result of Lenstra (see Subsection 1.3.3). This technique has also proven useful to obtain fixed-parameter tractability results with respect to the number of candidates for control in elections [99]. Hence, our result can be considered as a further example for the applicability of this method to voting.

For Manipulation, which is the special case of Possible Winner where the

input consists of a set of linear votes and a set of completely unordered votes, there is a broad study of the computational complexity in case of a bounded number of candidates and weighted votes [65]. Regarding scoring rules, this study shows that, for Borda and veto, Manipulation is solvable in polynomial time for up to two candidates and NP-complete for at least three candidates. In independent work Hemaspaandra and Hemaspaandra [129] obtained a full dichotomy which basically proves NP-hardness for all scoring rules except plurality. For plurality Manipulation is solvable in polynomial time for any number of candidates [65, 129]. Since the corresponding results do not imply NP-hardness for the unweighted voter case, the NP-hardness does not carry over to the Possible Winner problem considered in this work. Another recent work also makes use of scenarios in which the number of candidates is bounded: Xia et al. [199] state a polynomial-time algorithm with a "certain performance guarantee" for the optimization variant of Manipulation (as defined by Zuckerman et al. [204]). More specifically, for scoring rules, Xia et al. provide an algorithm such that the number of "additionally" needed manipulators is bounded by the number of candidates of the input instance.

Number of votes. Situations with few voters and a large number of candidates comprise for example meta-search or the selection of employees by agents in a human resource department. Since some of the voters might already provide linear votes, we usually partition the set of votes into proper partial and into linear votes. This directly leads to the three parameterizations "number of partial votes", "number of linear votes", and "total number of votes". Clearly, parameterized hardness results with respect to the "total number of votes" also hold for the two other parameterizations, whereas algorithmic results with respect to the "number of partial votes" carry over to the "number of total votes". We consider the parameterization by the "number of partial votes" as particularly interesting since intuitively it captures the "hard" part of an instance. This is also motivated by studies in the literature for Manipulation. In the Manipulation problem an instance consists of a set of linear votes and a set of completely unordered votes, called *coalition*. The size of the coalition is the number of partial votes. Constructive manipulation with bounded coalition size has been studied in several works, for example, [102, 200, 204]. Regarding scoring rules and unweighted voters, Xia et al. [199] showed NP-hardness for a scoring rule of very specific type for coalition size two.

We settle the computational complexity of Possible Winner for two specific voting systems in case of a constant number of votes. More specifically, we show that Possible Winner for k-approval is NP-hard for an instance consisting of at least two partial votes (and no linear votes) and Possible Winner for Borda is NP-hard for three partial votes and three linear votes. The corresponding many-one reductions are the technical main contributions of this chapter. Note that in the Chapter 9 we further refine the study of the parameter "number of partial votes" for k-approval: We investigate the combined parameter "number of partial votes" and k as well as the combined parameter "number of partial votes" and $m - k$ with m denoting the number of candidates.

Parameters measuring incompleteness. Measures for the amount of incompleteness of an instance are meaningful problem-specific parameterizations for Possible

Winner. Two natural parameterizations in this regard are the "number of undetermined pairs per vote" and the "total number of undetermined pairs". Regarding the first parameter, Xia and Conitzer [194] showed NP-hardness for constant parameter values for a broad class of scoring rules including Borda (see Section 7.5 for more details). The given many-one reductions only work for scoring rules with at least four different scoring values and thus do not apply to k-approval. We extend these results by providing an additional NP-hardness result for k-approval for a constant number of undetermined pairs per vote. These results imply that solely restricting the amount of incompleteness per vote does not decrease the computational complexity in terms of parameterized algorithmics for the considered scoring rules.

On the positive side, we develop fixed-parameter algorithms with respect to the parameter u denoting the "total number of undetermined candidate pairs". More specifically, we give a simple search tree algorithm of size at most 2^u working for all scoring rules and show how to improve the search tree size for a class of scoring rules including Borda and k-approval. Herein, the crucial part is the identification of a polynomial-time solvable special case. The positive results for the parameter "total number of undetermined candidate pairs" give evidence that a restricted amount of incompleteness can allow for efficient algorithms. Since in case of complete information the determination of a (possible) winner for scoring rules is trivial, this parameter can be understood as measuring the distance from triviality [124, 171, 172].[1] However, a drawback of the parameterization is that the parameter values are likely to be huge for many instances. Hence, at the end of this chapter, we introduce and discuss further parameterizations leading to challenges for future work.

8.1 Number of candidates

To assess the parameterized complexity with respect to the parameter "number of candidates", we employ Lenstra's famous algorithm for bounded-variable-cardinality integer linear programming (see Subsection 1.3.3). Lenstra's result says that it is fixed-parameter tractable with respect to the number of variables to check whether all inequalities of an integer linear program can be fulfilled at the same time. By providing an integer linear program formulation where the number of variables is bounded by a function only depending on the number of candidates, we obtain the following.

Theorem 8.1. *For all scoring rules,* Possible Winner *is fixed-parameter tractable with respect to the parameter "number of candidates".*

Proof. Let (C, P, c) be an input instance of Possible Winner with $m := |C|$. Consider a partial vote $v_i \in P$. Since there are $m!$ different linear orders on C, there are at most $m!$ possible linear orders that extend v_i. In case that v_i is already a linear order, there is exactly one extension, that is, v_i itself. In general, the same partial vote can occur more than once in P and each occurrence may be extended in a different way. Let $v'_1, \ldots, v'_q$ be the different partial votes in P and let n_i denote the number of occurrences of v'_i in P. For each v'_i, $i = 1, \ldots, q$, let $v_i^1, \ldots, v_i^{r_i}$ be all possible linear orders that extend v'_i with r_i denoting the number of linear orders extending v_i. Finally, let $s(d, v_i^j)$ denote the score that is assigned by the scoring rule to candidate d in the linear vote v_i^j.

[1] In this framework triviality refers to polynomial-time solvability.

Now, we can describe the integer linear program. Our set of variables is $\{x_{ij} \mid i \in \{1, \ldots, q\}$ and $j \in \{1, \ldots, r_i\}\}$. Intuitively, x_{ij} stands for the number of partial votes of type v'_i that are completed into the linear order v_i^j. We formulate Possible Winner as the feasibility problem for the following integer linear (in)equations:

$$\forall i \in \{1, \ldots, q\} : \quad \sum_{j=1}^{r_i} x_{ij} = n_i \tag{8.1}$$

$$\forall d \in C \setminus \{c\} : \quad \sum_{i=1}^{q} \sum_{j=1}^{r_i} x_{ij} \cdot (s(c, v_i^j) - s(d, v_i^j)) > 0 \tag{8.2}$$

The correctness can be seen as follows. The constraints (8.1) make sure that for every different partial vote v'_i the number of occurrences of v'_i in P equals the number of extensions. The constraints (8.2) model that the distinguished candidate c has to defeat all other candidates. Altogether, this implies that c is a possible winner if and only if the above constraints are feasible.

To make use of Lenstra's theorem, it remains to show that the number of variables is bounded by a function only depending on the number of candidates. First, we give a simple bound on the number of different partial votes and thus on q. Every partial vote v can be described by a set of ordered pairs of candidates, that is, (c', c'') is part of the set describing v if and only if $c' \succ c''$ in v. Since there are $m(m-1)$ ordered pairs and every partial vote is described by a proper subset of them, there are less than $2^{m(m-1)}$ types of partial votes. Second, r_i is clearly at most $m!$ (in case that one considers all extensions of an "empty" partial vote). Hence, there are less than $2^{m(m-1)}m!$ variables used in our formulation, making Lenstra's result applicable. □

Replacing ">" by "≥" in (8.2) only requires that c must make as least as much points as every other candidates and hence directly gives the analogous result for the co-winner case.

8.2 Number of votes

We investigate how the number of votes influences the computational complexity of Possible Winner for k-approval and Borda. Our main results are NP-hardness proofs for a constant total number of votes for both scoring rules. Before giving the corresponding many-one reduction, we make the following observation which will be used to establish a small dichotomy result in the case of k-approval.

Observation 8.1. *For every scoring rule,* Possible Winner *is solvable in polynomial time if the input instance contains at most one partial vote and an arbitrary number of linear votes.*

To see Observation 8.1, an algorithm deciding about the existence of a winning extension for the case of one partial vote can be sketched as follows. Basically, the algorithm tries out all possibilities to place c in the partial vote and then for each possibility checks whether filling the remaining positions from left to right by the remaining candidates can be done without beating c. In contrast, for two partial

votes the situation seems to become much more complicated because then different placements of a candidate in the partial votes might lead to the same score.

8.2.1 k-approval

We show that Possible Winner for k-approval becomes NP-complete for input instances consisting of at least two partial votes. Herein, we assume that k can be chosen appropriately within the given many-one reductions. The case that k as well as the number of votes are bounded will be considered in Chapter 9. The following NP-hardness results rely on many-one reductions from the NP-complete Independent Set (IS) problem. Given an undirected graph $G = (U, E)$ and a positive integer t, it asks whether there is a size-t vertex subset $U' \subseteq U$ such that there is no edge between any two vertices of U'.

Theorem 8.2. *For k-approval,* Possible Winner *is NP-complete for a partial profile that consists of at least two partial votes when k is part of the input.*

Proof. The NP-membership of Possible Winner for k-approval is obvious. To show the NP-hardness we give a many-one reduction from Independent Set. Consider an IS-instance $(G = (U, E), t)$ with n-vertex set $U = \{u_1, \ldots, u_n\}$[2]. We assume that $t < (n-1)/2$. This case clearly still leads to NP-completeness. We construct a partial profile P over a set C of candidates in which the distinguished candidate $c \in C$ is a possible unique winner according to k-approval with

$$k := n + \binom{n}{2} + |E| - tn + \binom{t}{2} + 1$$

if and only if G has an independent set of size at most t. Observe that the assumption $t < (n-1)/2$ implies that $k > 0$, and thus k-approval is well-defined. We first give a construction to show the hardness for a 2-voter profile and then explain how to extend this construction to work for an arbitrary fixed number of votes. The set of candidates is

$$C := \{c\} \uplus C_V \uplus C_E \uplus D$$

with

- one candidate for every vertex, that is, $C_V := \{c_i \mid v_i \in V\}$, and
- candidates related to unordered pairs of vertices, that is, for $i, j \in \{1, \ldots, n\}$, $i \neq j$,
  if $\{v_i, v_j\} \in E$, then there are two candidates $e_{\{i,j\}}$ and $e'_{\{i,j\}}$ in C_E;
  otherwise, there is one candidate $e_{\{i,j\}}$ in C_E.[3]
- The set D consists of dummy candidates with $|D| := n - t + \binom{n}{2} + |E| - tn + \binom{t}{2}$. Since $t < (n-1)/2$, this gives a positive integer.

The basic idea to construct the two partial votes (given in Figure 8.1) can be described as follows. The distinguished candidate is fixed such that it makes two

[2] Since the number of votes is constant, we use the variable n for the number of vertices.
[3] Note that $e_{\{i,j\}} = e_{\{j,i\}}$ and $e'_{\{i,j\}} = e'_{\{j,i\}}$.

$$
\begin{aligned}
v_1: &\quad c \succ D \succ C_V \succ C_E. \\
v_2: &\quad c \succ C_V \cup C_E \succ D, \text{ and, for } 1 \leq i < j \leq n, \\
&\quad \text{if } \{v_i, v_j\} \in E, \text{ then } c_i \succ e_{\{i,j\}} \text{ and } c_j \succ e'_{\{i,j\}}; \\
&\quad \text{otherwise,} \qquad\qquad c_i \succ e_{\{i,j\}} \text{ and } c_j \succ e_{\{i,j\}}.
\end{aligned}
$$

Figure 8.1: Partial votes of a Possible Winner instance resulting from a many-one reduction from Independent Set.

points in total. Thus, in a winning extension every other candidate must be assigned to a zero-position in at least one of the two votes. The first vote is used to select a set C'_V of t "vertex candidates". More specifically, we discuss below that k is adjusted such that exactly t vertex candidates from C_V (forming C'_V) must be assigned to one-positions whereas the remaining candidates from C_V and all candidates from C_E must be assigned to zero-positions and thus are beaten by c. It follows that the t candidates from C'_V must take a zero-position in the second vote. The second vote is constructed such that assigning a candidate from C_V to a zero-position implies that $n-1$ candidates from C_E are shifted to a zero-position (this is redundant in the sense that these candidates are already beaten by c since they assume zero-positions in the first vote). Now, the crucial part is that in a winning extension the t candidates from C'_V cannot shift pairwisely disjoint subsets of candidates from C_E since then the number of zero-positions would not be sufficiently large and a candidate from C'_V would be assigned to a one-position and thus would not be beaten by c. Clearly, if two candidates from C'_V shift a "common" candidate from C_E, then this "saves" one zero-position. The construction of the second vote ensures that two candidates from C_V can shift a common candidate from C_E to a zero-position only if there is no edge between them. Due to an appropriate bound on the number of zero-positions, this enforces that in a winning extension every pair of candidates from C'_V must shift a common candidate from C_E and thus there cannot be an edge between the corresponding vertices in the Independent Set-instance.

Claim: There is a winning extension of P if and only if G has a independent set of size t.

"$\Leftarrow$": Let $C_V^I \subset C_V$ denote the subset of candidates corresponding to an independent set of G. Then, the "edge candidates" that must be shifted to the right by the candidates from C_V^I are

$$
\begin{aligned}
C_E^I := &\{e_{\{i,j\}} \in C_E \mid c_i \in C_V^I \text{ and } \{v_i, v_j\} \notin E\} && (8.3)\\
\cup &\{e_{\{i,j\}} \in C_E \mid c_i \in C_V^I, i < j \text{ and } \{v_i, v_j\} \in E\} && (8.4)\\
\cup &\{e'_{\{i,j\}} \in C_E \mid c_i \in C_V^I, i > j \text{ and } \{v_i, v_j\} \in E\}. && (8.5)
\end{aligned}
$$

Before giving a winning extension of the partial votes, we verify that

$$
|C_E^I| = t \cdot (n-1) - \binom{t}{2}. \tag{8.6}
$$

For every candidate $c_i \in C_V^I$, C_E^I contains either the candidate $e_{\{i,j\}}$ or the candidate $e'_{\{i,j\}}$, for all $j \in \{1, \ldots, n\} \setminus \{i\}$, leading to the term $t \cdot (n-1)$. From this term we need to subtract the number of candidates that are counted twice. Since C_V^I

$$\begin{array}{lll} v_1: & c > D > C_V^I > & C_V \setminus C_V^I > C_E \\ v_2: & c > C_V \setminus C_V^I > C_E \setminus C_E^I > & C_V^I > C_E^I > D \end{array}$$

Figure 8.2: Winning extension of the partial votes with C_V^I denoting the candidates corresponding to an independent set and $C_E^I \subseteq C_E$ the candidates shifted to the right by candidates from C_V^I. The zero-positions are highlighted.

corresponds to an independent set, there are no edges between any pair of vertices corresponding to $\{c_i, c_j\} \subset C'_V$ and thus there is no candidate e'_{ij}. Hence, for every unordered pair of candidates $\{c_i, c_j\} \subset C'_V$, the candidate $e_{\{i,j\}}$ is counted twice, giving the term $\binom{t}{2}$.

Extend the partial votes as described in Figure 8.2. To see that c wins in this extension, we verify that the highlighted positions are the zero-positions. To this end, we first compute the number of zero-positions per vote which is the number of candidates minus k and thus

$$\begin{aligned} &1 + |C_V| + |C_E| + |D| - k = \\ &1 + n + \tbinom{n}{2} + |E| + n - t + \tbinom{n}{2} + |E| - tn + \tbinom{t}{2} - (n + \tbinom{n}{2} + |E| - tn + \tbinom{t}{2} + 1) = \\ &\qquad \binom{n}{2} + |E| + n - t. \end{aligned} \tag{8.7}$$

In v_1, the number of candidates taking the highlighted positions is $|C_E| + |C_V| - |C_V^I| = \binom{n}{2} + |E| + n - t$. In v_2, using Equation (8.6) one can verify that

$$\begin{aligned} |C_V^I| + |C_E^I| + |D| &= t + t(n-1) - \binom{t}{2} + n - t + \binom{n}{2} + |E| - tn + \binom{t}{2} \\ &= \binom{n}{2} + |E| + n - t. \end{aligned}$$

Since every candidate except c takes a highlighted position and thus a zero-position in at least one vote, it directly follows that c wins.

"$\Rightarrow$": Consider an extension in which c wins. By construction, in every extension of v_1, the last $|C_E| = \binom{n}{2} + |E|$ zero-positions must be assumed by the candidates from C_E. Due to Equation (8.7) there remain $n - t$ zero-positions which can only be assigned to candidates from C_V. Let $C_V^\star$ be the set of the remaining t candidates from C_V with a one-position in v_1. Let $C_E^\star \subseteq C_E$ be defined analogously to C_E^I by replacing C_V^I through $C_V^\star$ in (8.5). In a winning extension, all candidates from $C_V^\star$ must assume a zero-position in v_2. This shifts all candidates from $C_E^\star$ to zero-positions in v_2 as well. Since all candidates from D must assume zero-positions in every extension of v_2, it is easy to verify that there are exactly $tn - \binom{t}{2}$ zero-positions left over for the candidates from $C_V^\star$ and $C_E^\star$ (and c beats all candidates of D). In addition, a candidate from $C_E^\star$ can be shifted to a zero-position by at most two candidates from $C_V^\star$. Hence, there must be $\binom{t}{2}$ "shared" candidates in C'_E, that is, candidates $e_{\{i,j\}}$ with $c_i \succ e_{\{i,j\}}$ and $c_j \succ e_{\{i,j\}}$ in the partial vote v_2. Since $|C_V^\star| = t$, there are $\binom{t}{2}$ unordered pairs of candidates and each pair $\{c_i, c_j\}$ can share the candidate e_{ij} only if there is no edge between v_i and v_j in G (otherwise, the one of them with smaller index would

additionally shift $e'_{\{i,j\}}$). Hence the vertices corresponding to $C^\star_V$ form an independent set in G. This finishes the proof of the Claim.

Finally, let us consider the problem for $s > 2$ votes. To show NP-hardness, one pads the construction for two votes as follows. Add $(s-2) \cdot k$ new dummy candidates and fix them at the first k positions in $s-2$ of the votes such that every new candidate takes a one-position exactly once. The remaining two votes are constructed as in the 2-voter profile (given in Figure 8.1) and the new candidates are appended at the end. Clearly, every new candidate makes exactly one point and thus is always beaten by c. All old candidates make zero points in the newly added votes and thus the situation in the two "old" votes is still the same. Hence, c is a possible winner in the new s-voter profile if and only if it is a possible winner in the 2-voter profile. □

The construction given in the above NP-hardness proof can be adapted in a straightforward way to work for the co-winner case. Finally, combining Observation 8.1 with Theorem 8.2 one directly obtains the following dichotomy result.

Corollary 8.1. *For k-approval,* Possible Winner *with k being part of the input is NP-complete if the input profile consists of at least two partial votes and it is solvable in polynomial time otherwise.*

8.2.2 Borda

After considering Possible Winner for k-approval and a constant number of votes, we now investigate the same setting for the Borda rule. We first show that Possible Winner for Borda is NP-complete for a profile in which the number of partial votes is at least three making use of an unbounded number of linear votes. After this, we discuss how to obtain NP-hardness on profiles for a constant number of linear votes for specific constant numbers of partial votes. The results of this section are based on a many-one reduction from a special case of the 3-Partition problem.

> 3-Partition
> **Given:** A multi-set $A = \{a_1, \ldots, a_n\}$ of positive integers and $B := (3/n) \cdot \sum_{a_i \in A} a_i$.
> **Question:** Is there a partition of A into size-3 subsets $A_1, \ldots, A_{n/3}$ such that $\sum_{a_i \in A_j} a_i = B$ for each $j \in \{1, \ldots, n/3\}$?

Note that in this subsection n does not denote the number of votes (which is constant) but, following the literature, the number of integers from the 3-Partition instance. The 3-Partition problem is strongly NP-complete [118]. This implies that the NP-hardness still holds when the integers of A are polynomially bounded in n. We denote the special case that each integer $a_i \in A$ must be a multiple of n as 3-n-Partition. The 3-n-Partition problem is strongly NP-hard since every 3-Partition instance can be easily reduced to a 3-n-Partition instance by multiplying all input integers with n. Now, we provide the main result of this section.

Theorem 8.3. *For Borda,* Possible Winner *is NP-complete for an instance with partial profile $V^l \cup V^p$ if $|V^p| \geq 3$ with V^l being the set of linear and V^p being the set of partial votes.*

Proof. Let $A = \{a_1, \ldots, a_n\}$ denote a 3-n-Partition instance with $B := (3/n) \cdot \sum_{a_i \in A} a_i$. To ease the presentation, we first give a construction for the case that $a_i < a_{i+1}$ for $i = 1, \ldots, n-1$ and $a_1 = n$ and after this discuss the general case. We construct a partial profile $P = V^p \cup V^l$ with $|V^p| = 3$ over a set C of candidates in which the distinguished candidate $c \in C$ is a possible unique winner if and only if A is a yes-instance for 3-n-Partition. The set of candidates is

$$C := \{c\} \uplus E \uplus T \uplus D,$$

with one candidate for every member of A, that is, $E := \{e_i \mid a_i \in A\}$, candidates representing the subsets resulting from the partition into 3-sets, that is, $T := \{t_1, \ldots, t_{n/3}\}$, and a set of dummy candidates $D := \biguplus_{i=1}^{n} D_i$ only needed to "fill" positions and further specified in the following.

The set of partial votes V^p consists of three identical partial votes v_1, v_2, and v_3. Each of them is defined as follows:

$$c \succ T \text{ and } c \succ D_1 \succ e_1 \succ D_2 \succ \cdots \succ D_i \succ e_i \succ \cdots \succ D_n \succ e_n$$

with $|D_1| = a_1 - 1$ and $|D_i| = a_i - a_{i-1} - 1$ for $i \in \{2, \ldots, n\}$. This definition fixes the number of dummy candidates; more precisely,

$$|D| = |D_1| + |D_2| + \cdots + |D_n| = a_1 - 1 + \sum_{i=2}^{n} (a_i - a_{i-1} - 1) = a_n - n.$$

Thus, the total number of candidates is

$$m = 1 + |E| + |T| + |D| = 1 + n + n/3 + a_n - n = a_n + n/3 + 1.$$

Since 3-n-Partition is *strongly* NP-complete, we can assume that a_n and, thus, m is polynomial in n. Having a closer look at the partial votes, for every candidate e_i corresponding to an integer a_i from A, the following holds within every partial vote:

$$|\{s \in C \setminus T : s \succ e_i\}| = a_1 - 1 + \sum_{j=2}^{i} (a_j - a_{j-1} - 1) + i - 1 + 1 = a_i. \tag{8.8}$$

Furthermore, the position and thus the total score of the distinguished candidate c is already fixed. In contrast, every subset candidate $t_j \in T$ can be "inserted" at any position behind c in the three partial votes. The basic idea of this construction is that the "choice" of the positions for t_j in the three partial orders corresponds to the choice of three numbers from A into the corresponding subset A_j. For example, inserting t_j directly before the candidate e_i in one of the partial votes means that $a_i \in A_j$. To this end, our goal is to ensure the following two properties for every winning extension of P:

- Every number of A is *selected* exactly once, that is, for every candidate $e_i \in E \setminus \{e_1\}$ there is exactly one candidate $t_j \in T$ with "$e_{i-1} > \cdots > t_j > \cdots > e_i$" within one of the three extended votes from V^p, and one candidate $t_j \in T$ with "$t_j > \cdots > e_1$".
- For every $t_j \in T$, the sum corresponding to the three "number candidates" from E selected by t_j is B.

As argued in the remainder of the proof, these two points can be realized by setting the linear orders V^l such that the following maximum partial scores hold (using Lemma 7.1).

- $s_p^{\max}(e_i) = 3(m-1) - 3a_i - i$ for all $e_i \in E$,
- $s_p^{\max}(t_j) = 3(m-1) - B$ for all $t_j \in T$, and
- $s_p^{\max}(d) \geq 3(m-1)$ for all $d \in D$.

This implies that a candidate $d \in D$ cannot beat c in any extension. For the other candidates, we interpret the maximum partial scores as follows. Since the maximum amount of points a candidate can make within an extension of the partial votes is $3(m-1)$, the maximum partial scores also provide the number of points a candidate must *loose* to be beaten by c. More specifically, every $e_i \in E$ must loose at least $3a_i + i$ points and every $t_j \in T$ must loose at least B points. Herein, a candidate looses one point for every other candidate that is placed before it in an extension of one of the partial votes. Now, we show the following.

Claim: There is a solution for 3-n-Partition if and only if there is an extension of P such that c wins.

"$\Rightarrow$": Let $\{A_1, \ldots, A_{n/3}\}$ with $A_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$ denote a solution of 3-n-Partition for A. Then, extend v_1 such that "$D_{j_1} > t_j > e_{j_1}$" for every $j \in \{1, \ldots, n/3\}$, v_2 such that "$D_{j_2} > t_j > e_{j_2}$" for every $j \in \{1, \ldots, n/3\}$, and v_3 such that "$D_{j_3} > t_j > e_{j_3}$" for every $j \in \{1, \ldots, n/3\}$; that is, every candidate t_j is inserted directly before the three candidates corresponding to the integers from A_j. Since all integers from A are pairwisely distinct, this extension is unambiguous. In every partial vote, for every $e_i \in E$, there are exactly a_i candidates $s \in C \setminus T$ with $s \succ e_i$ (see Equation (8.8)). Thus, without inserting any $t_j \in T$ before e_i, e_i looses $3a_i$ points. For $q \in \{1, 2, 3\}$, let $\tau_{i,q}$ denote the number of candidates from T that are inserted before e_i in the considered extension of v_q. Then, for every e_i, we have $\tau_{i,1} + \tau_{i,2} + \tau_{i,3} = i$ since for each $z \in \{1, \ldots, i\}$ a candidate from T is inserted directly before e_z in one of the three partial votes. Thus, e_i looses $3a_i + i$ points in this extension and c beats e_i. It remains to show that c beats t_j for each $t_j \in T$. Since $a_{j_1} + a_{j_2} + a_{j_3} = B$, analogously to Equation (8.8), one can compute that the number of candidates that are "better" than t_j in the three partial votes is at least

$$\begin{aligned} |\{s \in C \setminus T \mid s \succ e_{j_1} \text{ in } v_1\}| &\quad + \\ |\{s \in C \setminus T \mid s \succ e_{j_2} \text{ in } v_2\}| &\quad + \\ |\{s \in C \setminus T \mid s \succ e_{j_3} \text{ in } v_3\}| &\quad = B. \end{aligned}$$

It follows that within the considered extension t_j makes at most $3(m-1) - B = s_p^{\max}(t_j)$ points and thus is beaten by c.

"$\Leftarrow$": Consider a winning extension of the partial votes. As explained before without "counting" any $t_i \in T$ before e_i, e_i "looses" $3a_i$ points (see Equation (8.8)). Hence, in every winning extension at least i times a candidate from T must be inserted before e_i, that is,

$$\tau_{i_1} + \tau_{i_2} + \tau_{i_3} \geq i.$$

We denote this as *selection property* (which will be crucial in the following argumentation). For every winning extension, we show the following two properties (also stated above):

1. For every candidate $t_j \in T$ selecting $e_{j_1}, e_{j_2}, e_{j_3}$, one must have $\sum_{q=1}^{3} a_{j_q} = B$.
2. Every candidate from E, that is, every number of A, is selected exactly once.

To show the first property, we devise a proof by contradiction showing that one can neither have $\sum_{q=1}^{3} a_{j_q} < B$ nor $\sum_{q=1}^{3} a_{j_q} > B$. Assume that in a winning extension there is a t_j selecting three candidates $e_{j_1}, e_{j_2}, e_{j_3}$ such that $\sum_{q=1}^{3} a_{j_q} < B$. Then, the minimum number of points that t_j makes in this extension is $3m - 3$ minus the number of candidates which are placed before t_j in this extension (summing over all three votes). In V_q, $q \in \{1, 2, 3\}$, due to Equation (8.8) at most a_{j_q} candidates of $C \setminus T$ can be placed before t_j since otherwise t_j would not have selected e_{j_q} (but a candidate from E which is right from e_{j_q} for at least one q). Since $|T \setminus \{t_j\}| = n/3 - 1$ and every candidate from $T \setminus \{t_j\}$ can be inserted at most once before t_j in every partial vote, t_j must make at least

$$3m - 3 - \sum_{q=1}^{3} a_{j_q} - n + 3 = 3m - n - \sum_{q=1}^{3} a_{j_q}$$

points. By assumption, $\sum_{q=1}^{3} a_{j_q} < B$ and since all $a_{j_q} \in A$ are multiples of n, this means that $\sum_{q=1}^{3} a_{j_q} \leq B - n$. Then, in total t_j will make at least

$$3m - n - B + n = 3(m - 1) - B + 3 > s_p^{\max}(t_j)$$

points in the partial votes, and t_j thus beats c, a contradiction.

Now, assume that there is a t_j with $\sum_{q=1}^{3} a_{j_q} > B$ in a winning extension. Consider the amount of points all remaining candidates from $T \setminus \{t_j\}$ together can loose by candidates from $C \setminus T$. Due to the selection property, at least i candidates from T must be inserted before every $e_i \in E$. Clearly, inserting every candidate from $T \setminus \{t_j\}$ as far right as possible (that is, directly before the selected candidate) maximizes the amount of points the candidates from $T \setminus \{t_j\}$ can loose. Using Equation (8.8) and further using the assumption that $\sum_{q=1}^{3} a_{j_q} > B$, this amount is

$$\sum_{i=1}^{n} a_i - \sum_{q=1}^{3} a_{j_q} = (n/3) \cdot B - \sum_{q=1}^{3} a_{j_q} < (n/3 - 1) \cdot B.$$

This amount can be "contributed" to the candidates only in multiples of n since a_i differs from a_j at least by n. Hence, at least one candidate t of the $n/3 - 1$ candidates from $T \setminus \{t_j\}$ must loose less than B points and, thus, can loose at most $B - n$ points by candidates from $C \setminus T$. Again, t can loose at most $n - 3$ additional points by inserting other candidates from T before it. Thus, the minimum score that t will make is

$$3(m - 1) - B + n - n + 3 > s_p^{\max}(t)$$

and t will beat c, a contradiction.

To see the second property, it remains to show that every number in A is selected exactly once. We give a proof by induction: The "last" number candidate e_n cannot be selected twice (or more times) since this would imply that at most $n - 2$ times a

candidate from T could be inserted before e_{n-1} and this would violate the selection property. Now, for any $i \in \{1, \ldots, n-1\}$ consider e_i and assume that every $e_j, j > i$, has been selected exactly once. Then, selecting e_i twice (or more times) implies that at most $n - 2 - (n - i) = i - 2$ times a candidate from T can be inserted before e_{i-1}, again violating the selection property.

Summarizing, we have shown that in every winning extension, every candidate from E is selected exactly once and every candidate from T selects three candidates whose corresponding numbers sum up to B. Hence, the subsets $A_j := \{a_{j_1}, a_{j_2}, a_{j_3} \mid t_j \text{ selects } e_{j_1}, e_{j_2}, e_{j_3}\}$ for $j \in \{1, \ldots, n/3\}$ form a solution of the 3-n-Partition instance. This finishes the proof of the Claim.

Now, we briefly discuss how to modify the construction if the numbers from A are not pairwisely different. Then, the candidate subset E consists of one candidate for every different number in A and we want to have that a candidate $e_i \in E$ representing s equal numbers is selected s times. This is achieved by adapting the maximum partial score of e_i as follows:

$$s_p^{\max}(e_i) = 3(m-1) - 3a_i - |\{a_j \in A \mid a_j \geq a_i\}|.$$

Then, the proof for this case works in complete analogy to the given proof for the special case.

Finally, for every fixed number $s > 3$ of partial votes, one can apply an analogous reduction from s-n-Partition (or allow to fix all but three partial votes and adapt the maximum partial scores appropriately). □

The reduction used for the proof of Theorem 8.3 can be adapted to the co-winner case by decreasing the maximum partial score of every non-distinguished candidate by one.

Number of linear votes. Theorem 8.3 makes use of Lemma 7.1 to construct a multiset of linear votes leading to the required maximum partial scores. We briefly discuss how to find an equivalent constant-size set of linear votes for some cases. To this end, it is not hard to construct three linear votes that "realize" the maximum partial scores for the candidates from A and E as defined within the proof (for exactly three partial votes). For example, one can set c such that it makes exactly one point in total and, then, set e_i to position a_i in two of the votes and to position $a_i + i$ in the third vote. It remains to place the candidates from D such that they are beaten by c in every extension. This can be done without adding further candidates by rearranging them in a sophisticated way (see [133]). Another possibility is to append a set D' of further "dummy" candidates with $|D'| = |D|$ which is used to fill the remaining positions "between" the candidates from $\{c\} \cup A \cup E$ in the three linear votes whereas the candidates from D are appended at the end of the linear votes. Clearly, appendig the candidates from D, changes the total scores of the other candidates in the partial votes but does not affect their relative differences. Then, modifying the partial votes by appending the candidates from D' at the end (in arbitrary order) yields a construction for three linear and three partial orders as needed for the NP-hardness proof.

Corollary 8.2. *For Borda,* Possible Winner *is NP-complete for a partial profile consisting of three partial and three linear votes.*

Although it seems plausible that similar constructions can be given for other fixed numbers of linear and partial votes such as having an instance with four partial and three linear votes, this seems laborious and not of much interest in general.

Finally, we remark that in contrast to k-approval, we do not provide a full dichotomy for Borda. In particular, for input instances containing exactly two partial votes (and some additional linear votes), the computational complexity is unsettled.

8.3 Measures of incompleteness

After having studied the two standard voting parameters "number of candidates" and "number of votes", this section is concerned with problem-specific parameterizations for Possible Winner. We introduce parameters measuring the "amount of incompleteness" of an instance. A natural way to do so is to consider the "number of undetermined pairs".

Definition 8.1. For a partial vote, two candidates c_1 and c_2 form an *undetermined pair* if neither $c_1 \succ c_2$ nor $c_2 \succ c_1$ is part of the vote.

This definition directly leads to two parameterization studied in the following:

1. The maximum number of undetermined pairs per vote.
2. The total number of undetermined pairs of an instance.

For the first parameter, there are NP-hardness results for constant parameter values for a class of scoring rules [194] (not comprising k-approval). We apply a new many-one reduction to show NP-hardness for k-approval for $k \in \{4, \ldots, m-4\}$. In contrast, for the second parameter, it is trivial to obtain fixed-parameter tractability for scoring rules in general. We further improve this trivial result for a class of scoring rules comprising Borda and k-approval. Herein, a crucial part is the identification of a polynomial-time solvable special case. A drawback of the second parameterization is that the parameter values seem to be quite large for many instances. Hence, at the end of this section we discuss further parameterizations based on undetermined pairs (lying "between" the two considered parameterizations), leading to future research challenges.

8.3.1 Maximum number of undetermined pairs per vote

Xia and Conitzer [194] showed for several common voting rules that the Possible Winner problem is NP-complete even if each partial vote only contains a constant number of undetermined pairs. This comprises an NP-hardness result for a subclass of scoring rules including Borda in case that there are at least four undetermined pairs per vote. The corresponding many-one reductions and class of scoring rules have been discussed in Section 7.5. The given reductions only work for scoring rules with at least four different scoring values and thus do not apply to k-approval. We apply a reduction from Exact Cover by 3-Sets to obtain the following result for k-approval.[4] Recall that, given a family $\mathcal{S}$ of subsets over an element set E, the X3C-problem asks whether

[4] Independently, Xia and Conitzer [197] showed NP-hardness for k-approval for fixed $k > 2$ even for four undetermined pairs per vote.

there is subset $\mathcal{S}'$ of $\mathcal{S}$ such that every element from E is contained in exactly one of the sets from $\mathcal{S}'$.

Theorem 8.4. *For k-approval with $4 \leq k \leq m-4$,* Possible Winner *is NP-complete for 16 undetermined pairs per vote.*

Proof. We first give a reduction from Exact Cover by 3-Sets to Possible Winner for 4-approval. Let $(E, \mathcal{S})$ be an X3C-instance. We construct an instance of Possible Winner that is a yes-instance if and only if $(E, \mathcal{S})$ is a yes-instance. The set of candidates consists of one candidate for every element from E, the distinguished candidate c, one candidate x and four further candidates s_1, s_2, s_3, and s_4. For every subset $S_i \in \mathcal{S}$, the set of partial votes V^p contains one partial vote defined as follows:

$$x \succ e_{i1} \succ e_{i2} \succ e_{i3} \succ C_i \text{ and } s_1 \succ s_2 \succ s_3 \succ s_4 \succ C_i$$

where e_{i1}, e_{i2}, e_{i3} denote the candidates corresponding to the three elements from S_i and the candidates from $C_i := C \setminus \{x, e_{i1}, e_{i2}, e_{i3}, s_1, s_2, s_3, s_4\}$ are fixed at an arbitrary order. Since only candidate pairs formed by one candidate $s_j, j \in \{1, 2, 3, 4\}$, and one candidate from $\{x, e_{i1}, e_{i2}, e_{i3}\}$ are not fixed, there are 16 undetermined pairs per vote. Let $k := |E|/3$. Using Lemma 7.1, we construct of set of linear votes such that

- $s_p^{\max}(x) = k$,
- $s_p^{\max}(s_i) = |V^p| - k$, and
- $s_p^{\max}(e) = 1$.

The correctness of the construction can be seen as follows. Consider a solution for the X3C-instance. Then, extend every vote corresponding to a subset S_i of the solution to

$$x > e_{i1} > e_{i2} > e_{i3} > s_1 > s_2 > s_3 > s_4 > \dots$$

and every remaining partial vote to

$$s_1 > s_2 > s_3 > s_4 > x > e_{i1} > e_{i2} > e_{i3} > \dots \ .$$

In this extension, the score of every candidate is exactly its maximum partial score and hence c is a winner.

Consider a winning extension. Due to its maximum partial score, the candidate x must assume a zero-position in at least $|V^p| - k$ of the votes. The only way to extend a vote such that x ends up at a zero-position is to insert all candidates $s_j, j \in \{1, 2, 3, 4\}$, before x. This implies that every candidate $s_j, j \in \{1, 2, 3, 4\}$, must make already $|V^p| - k$ points in the corresponding votes and thus must assume the zero-positions within the remaining k votes. Considering the remaining $k = |E|/3$ votes, every element candidate can be at a one-position at most once without beating c. Hence, the corresponding subsets must correspond to an exact 3-cover for $(E, \mathcal{S})$.

For $4 < k \leq m-4$ the reduction can be adapted either by padding the first positions by dummy candidates or, for values of k close to m, by "flipping" the construction by putting the undetermined pairs at the end of the partial votes. □

8.3.2 Total number of undetermined pairs

As a consequence of the NP-hardness results from the previous subsection, there is no hope for showing fixed-parameter tractability with respect to parameter "number of undetermined pairs per vote" for many scoring rules including Borda and k-approval. To chart the border of tractability, we consider the parameter u denoting the "total number of undetermined pairs", that is, the sum of the number of undetermined pairs over all votes. We first give a simple and general depth-bounded search tree showing fixed-parameter tractability with respect to u for all scoring rules. Then we give a faster algorithm for a subclass of scoring rules including Borda and k-approval. The improved algorithm relies on the identification of a special case which, although still being NP-hard for some scoring rules, can be solved in polynomial time for Borda and k-approval.

A general search tree approach

For every partial vote v and for every undetermined pair $\{c_i, c_j\}$ in v, we branch into the two possible cases by adding either $c_i \succ c_j$ or $c_j \succ c_i$ to v. If one of these options violates the transitivity of v, then discard this option. This directly yields a search tree of size at most 2^u. Clearly, all undetermined pairs can be found in $O(nm^2)$ time. For an arbitrary voting rule r, let $f_r(n, m)$ denote the running time needed to compute a winner when given linear orders. Then, for every leaf one can check whether c is a winner for the corresponding extension in $f_r(n, m)$ time, implying the following.

Proposition 8.1. *For a partial n-voter profile over m candidates and a voting rule r,* Possible Winner *can be decided in $O(2^u \cdot (m + f_r(n, m)) + nm^2)$ time, where u denotes the total number of undetermined pairs.*

Corollary 8.3. *For every scoring rule,* Possible Winner *is fixed-parameter tractable with respect to the parameter "total number of undetermined pairs".*

An improved search tree and a polynomial-time solvable special case

Similar to the improved search tree for Kemeny Score (see Section 3.4), for some scoring rules the general search tree can be improved by a more refined branching, that is, branching into "undetermined triples" instead of undetermined pairs. Three candidates $\{c_1, c_2, c_3\} \subseteq C$ form an *undetermined triple* with respect to some partial vote v if there are at least two undetermined pairs in v, each formed by two candidates from $\{c_1, c_2, c_3\}$. Branching on undetermined triples instead of undetermined pairs leads to a search tree of size 1.82^u. More precisely, for an undetermined triple consisting of three undetermined pairs one branches into all six possible orders and can decrease the parameter by three in every case (giving the branching vector $(3, 3, 3, 3, 3, 3)$ with branching number 1.82). Any other undetermined triple consists of two undetermined pairs and thus allow for three possible extensions resulting in the branching vector $(2, 2, 2)$ with branching number 1.72. The interesting part is now to investigate the situation after exhaustively branching into all undetermined triples. To formalize this, we introduce the following.

Definition 8.2. For a partial vote v, an undetermined pair of candidates is *isolated* if in v both candidates do not form an undetermined pair with any other candidate.

Since for an isolated undetermined pair $\{c_1, c_2\} \subseteq C$ of candidates, the relative order of c_1 and c_2 with respect to all other candidates is already determined, c_1 and c_2 must have the same relative order with respect to each of the remaining candidates. Thus, they must end up as direct neighbors in every extension. Now, we can use the following characterization to investigate the situation after branching into triples.

Observation 8.2. *Every undetermined pair must either be part of an undetermined triple or must be isolated.*

The correctness of Observation 8.2 can be seen as follows. Assume that there is an isolated undetermined pair $\{c_1, c_2\} \subseteq C$ which is part of an undetermined triple formed by c_1, c_2, and c_3. By definition, an undetermined triple must contain at least two undetermined pairs, that is, either $\{c_1, c_3\}$ or $\{c_2, c_3\}$ must form an undetermined pair. Then, $\{c_1, c_2\}$ cannot be isolated, a contradiction.

The refined branching strategy based on undetermined triples combined with Observation 8.2 directly leads to the following result.

Proposition 8.2. *Consider a scoring rule for which* Possible Winner *can be decided in polynomial time if all undetermined pairs are isolated. Then, for this scoring rule,* Possible Winner *can be solved in* $1.82^u \cdot poly(n, m)$ *time.*

Proposition 8.2 motivates the study of Possible Winner for scoring rules in instances in which all undetermined pairs are isolated. For this problem, we show NP-hardness for a natural class of scoring rules but provide polynomial-time algorithms for another class including Borda and k-approval. Since the proof of the following theorem is conceptually very similar to the proof of Lemma 7.4, we only sketch the proof.

Theorem 8.5. *Consider a scoring rule such that, for every number of candidates, the scoring vector* α *contains a position* g *such that* $\alpha_g \geq \alpha_{g+1} + 3x$ *and a position* h *such that* $\alpha_h = \alpha_{h+1} + x$ *for a positive integer* x*. Then,* Possible Winner *is NP-complete even if all undetermined pairs are isolated.*

Proof. (Sketch) We give a reduction from Exact Cover By 3-Sets to Possible Winner. Let $(E, \mathcal{S})$ denote an X3C-instance. We construct an instance (P, C, c) of Possible Winner that is a yes-instance if and only if $(E, \mathcal{S})$ is a yes-instance. The set of candidates consists of one candidate s_i for every $S_i \in \mathcal{S}$, one candidate for every element from E, a further candidate d, and the distinguished candidate c.

The set of partial votes V^p consists of two subsets. First, for every $S_i \in \mathcal{S}$, there is one vote in which the candidates d and s_i form an isolated pair which must end up at positions g and $g + 1$ with $\alpha_g \geq \alpha_{g+1} + 3x$. Second, for every $S_i \in \mathcal{S}$ and for every element $e \in S_i$, there is one partial vote in which s_i and the candidate corresponding to e form an isolated pair which is located at the positions h and $h + 1$ with $\alpha_h = \alpha_{h+1} + x$. The remaining candidates in all partial votes are fixed in an arbitrary order which "ensures" the proper placement of the specified isolated pairs.

Using Lemma 7.1, we can construct a set of linear votes such that in a winning extension one has the following.

- Every subset candidate s_i can make at most $\alpha_g + 3\,\alpha_{h+1}$ points in the four partial votes in which it is not fixed. In other words, if s_i takes the better position, that

is, position g, in the isolated pair formed with d, then it must take position $h+1$ in the three other corresponding isolated pairs. If it takes position $g+1$ in the pair formed with d, then it can still take position h for all three remaining pairs since $\alpha_g \geq \alpha_{g+1} + 3x$ and $\alpha_h = \alpha_{h+1} + x$.

- Every candidate corresponding to E must assume position $h+1$ for at least one isolated pair.
- Candidate d must assume position $g+1$ at least $\mathcal{S} - |E|/3$ times.

Then, it is not hard to see that there is a winning extension if and only if there is an exact 3-cover: In the first set of partial votes, the candidate d can (and must) take position g in $|E|/3$ votes. Thus, d can be considered as to "select" the corresponding $|E|/3$ subset candidates. The corresponding subset candidates in these votes then assume position $g+1$ in this votes and thus have at least $3x$ points left which they can use to take the better position in the three votes in which they form isolated pairs with their element candidates. Then, the only possibility that every element assumes position h for at least one isolated pair is that it is part of a subset "selected" by d in the first set of partial votes. Hence, the selected subsets must correspond to an exact 3-cover. □

In the following, we complement Theorem 8.5 by providing a polynomial-time algorithm for a class of scoring rules including k-approval and Borda using network flows techniques.

Theorem 8.6. *Consider a scoring rule such that, for every number of candidates, the scoring vector α fulfills $\alpha_i \leq \alpha_{i+1} + 1$ for $i \in \{1, \ldots, m-1\}$ and let u denote the total number undetermined pairs. Then,* Possible Winner *can be decided in $O(nm^2 + u^2)$ time if all undetermined pairs are isolated.*

Proof. As a first step, compute the set of all (isolated) undetermined pairs. Clearly, this can be done in $O(nm^2)$ time. Recall that two candidates forming an undetermined pair must end up as direct neighbors in every extension. As a preprocessing step, fix the order of some "trivial" undetermined pairs. If the distinguished candidate c is contained in an undetermined pair, then fix the order of this pair such that c is placed in front of the second candidate of the pair. After that, one can assume that none of the undetermined pairs contains c. Furthermore, consider an isolated undetermined pair $\{c_i, c_j\}$ which must end up at positions z and $z+1$ with $\alpha_z = \alpha_{z+1}$. Then, the relative order of c_i and c_j does not change their total scores and hence we can fix such pairs in an arbitrary order.

For every remaining undetermined pair $\{c_i, c_j\}$, the difference of the score between choosing $c_i > c_j$ and choosing $c_j > c_i$ is exactly one point for every candidate. For every candidate $c_i \in C \setminus \{c\}$ one can compute the minimum number $l(c_i)$ of points that c_i will make in every possible extension. That is, $l(c_i)$ is the sum over the scores for c_i obtained by choosing $c_j \succ c_i$ in all undetermined pairs $\{c_i, c_j\}$ that contain c_i. Recall that the score $s(c)$ for the distinguished candidate c is already fixed. If $l(c_i) \geq s(c)$ for some c_i, then c cannot become possible winner. Otherwise, let $b(c_i) := s(c) - l(c_i) - 1 \geq 0$ denote the *balance* of c_i with respect to c. The balance counts the number of undetermined pairs where c_i can be placed better than the other respective candidate in a winning extension.

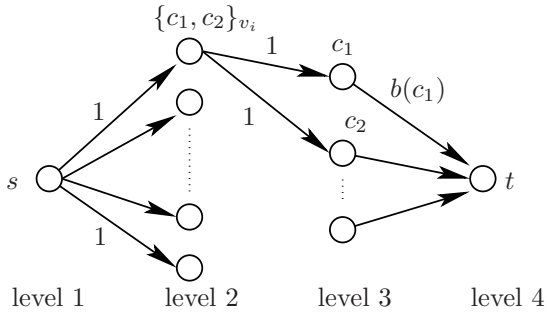


Figure 8.3: Flow network which allows for a flow of size u if and only if all undetermined pairs can be ordered such that the distinguished candidate wins in the resulting extension.

Using the balance $b(c_i)$ for all candidates $c_i \in C \backslash \{c\}$, one can decide Possible Winner with the help of a maximum flow computation as follows. Consider a four-level directed, arc-weighted s-t-network with distinguished vertices s and t (see Figure 8.3). The first level only consists of vertex s. The second level consists of vertices one-to-one representing all undetermined pairs. Note that an undetermined pair formed by the same two candidates may occur several times for different partial votes. The vertex s is connected by arcs of weight one to all level-two vertices. The third level of vertices one-to-one represents all candidates occurring in at least one undetermined pair. Every level-two vertex representing an undetermined pair is connected by two weight-one arcs to the two vertices corresponding to the two candidates contained in the undetermined pair. The fourth level only consists of vertex t. Every level-three vertex representing a candidate c_i is connected by one arc to t which is assigned the weight $b(c_i)$.

Claim: The constructed flow network allows for an integer flow of value u if and only if the distinguished candidate c is a possible winner of the corresponding Possible Winner-instance with u undetermined pairs.

"$\Leftarrow$": If u undetermined pairs of the Possible Winner instance can be resolved without defeating c, then one obtains the following network flow. Send one unit of flow through every arc between level one and level two. For every level-two vertex send this unit of flow to the vertex corresponding to the candidate ordered better in the isolated pair. For every level-three vertex, pass the resulting units straightforward to t. It is not hard to see that this flow is feasible: The only arcs through which the flow might send several units are going from the third to the fourth level. For such an arc corresponding to a candidate c_i there is exactly one unit of flow for each placement of c_i at the "better" position in an undetermined pair. Thus, a the corresponding flow consists of at most $b(c_i)$ units and thus is feasible.

"$\Rightarrow$": If the maximum integer flow of the network is u, then all u second-level vertices corresponding to the undetermined pairs get one unit of flow. Each of them has to pass this unit on to one of the two level-three vertices corresponding to the candidates forming the considered pair. Each "candidate vertex" c_i in level three sends at most $b(c_i)$ units of flow to t. Hence, putting c_i to the better position in the undetermined pairs that "contributed" to the flow through the "c_i-vertex" gives a winning

extension.

Finally, we show the overall running time $O(nm^2 + u^2)$. The flow network can be constructed in $O(nm^2)$ time. The number of arcs of the flow network is linear in u. The Ford-Fulkersson algorithm can compute a maximum integer flow in $O(|A| \cdot f)$ time, where $|A|$ denotes the number of arcs and f denotes the value of a maximum flow [66]. Since the value of the maximum flow is bounded by u, the claimed running time follows. □

Corollary 8.4. *For Borda and for k-approval,* Possible Winner *can be decided in $O(1.82^u \cdot (nm^2 + u^2))$ time, where u denotes the total number of undetermined pairs.*

Note that Theorem 8.5 and Theorem 8.6 do not give a complete classification. In particular, the computational complexity for the case that there are scoring vectors with "maximum" distance two between two consecutive scoring values remains unsettled.

8.3.3 Further parameters measuring incompleteness

In Subsection 8.3.1, we have seen that for a broad class of scoring rules there is no hope for fixed-parameter tractability with respect to the "number of undetermined pairs per vote". In contrast, the fixed-parameter tractability results with respect to the parameter "total number of undetermined pairs" show that being not too far from complete information makes the Possible Winner problem provably easier (8.3.2). However, this result suffers from the fact that the considered parameter may assume quite large values for many instances. This motivates the study of further parameterizations measuring the amount of incompleteness. In the following, we suggest some parameterizations and ideas that might serve as a basis for future research.

- Instead of measuring the amount of incompleteness per vote, it also seems plausible to measure the amount of incompleteness *per candidate*. This directly leads to the parameterizations "maximum/average number of undetermined pairs in which a candidate is involved".

- The parameterizations suggested in the previous subsection measure the amount of incompleteness. An "opposite" perspective is to measure the amount of *completeness* within the partial votes, for example, by parameterizations based on the "number of determined pairs". This directly leads to the "dual parameterizations" with respect to the parameterizations from the previous subsections. For example, the number of determined pairs per vote is the total number of pairs minus the number of undetermined pairs per vote. Such parameterizations can be considered as extending the special case of Manipulation where the number of determined pairs in every non-linear vote is zero.

- The development of a refined search tree in Section 8.3.2 led to the identification of the special case that all undetermined pairs of candidates are isolated. In this case, Possible Winner becomes solvable in polynomial time for a class of scoring rules including Borda and k-approval. The concept of isolated pairs can be considered as a measure of "local disturbance". Along these lines, it might be interesting to investigate instances allowing for isolated triples and,

more generally, isolated tuples of bounded size. However, since the size of an isolated tuple is smaller than two times the "number of undetermined pairs per vote", the hardness results from Subsection 8.3.1 can be transferred.

- An interesting parameterization concerns the "number of possible extensions per vote".[5] In general, the computation of this parameter, that is, counting the number of linear extensions for one partial vote is computationally hard [47]. Fortunately, the set $E(v)$ of extensions of a partial order v, can be generated in time constant in $|E(v)|$ [180]. Note that it is easy to see that this parameterization is fpt-equivalent to the parameter "number of undetermined pairs per vote". However, it might allow for a different view that is helpful to design algorithms.

Concluding, we think that the development and investigation of further parameterizations measuring the amount of (in)completeness is an important challenge for future research. Clearly, this is a general conceptual task also of interest for voting rules other than scoring rules.

The discussion above dealt with the identification of new single parameterizations that might lead to tractability for meaningful cases. In the following chapter, we present an alternative way to obtain fixed-parameter tractable cases by investigating combined parameters (where known single parameters lead to W[1]-hardness). More specifically, the usefulness of combined parameters still capturing meaningful scenarios will be exhibited using Possible Winner for k-approval voting as example.

[5] There are some recent considerations of the counting variant of Possible Winner [7].

Chapter 9

Combined parameters for k-approval

Under the k-approval rule every voter can assign one point to exactly k alternatives and an alternative with most points in total wins. In the case that every voter provides complete information, the winner can be easily determined. However, there are settings in which the voters may only provide partial information on their preferences. This directly leads to the central combinatorial problem considered in this part: the Possible Winner problem, which asks whether a specific alternative can still become a winner. In Chapter 7, we provided NP-completeness for every $k \in \{2, \ldots, m-2\}$ with m denoting the number of alternatives if the number of votes is unbounded. In Chapter 8, we showed that Possible Winner for k-approval is also NP-complete if there are only two partial votes (and k is part of the input).

These hardness results motivate a multivariate complexity analysis with respect to the combined parameter "number of votes" *and* "number of candidates to which a voter gives one/zero points" for k-approval. Can we efficiently solve Possible Winner when these parameters are both small? This setting might look restrictive on a first glance but it naturally reflects scenarios in which one is interested in finding a small group of winners (or losers). For example, a small committee awards a small number of grants or picks out a limited number of students for graduate school. Another example might appear in a human resource department where few people select few employees out of a large pool of job applicants. As concrete example one might look at the decision about the Nobel prize for peace in 2009, where a committee consisting of five people had to select up to three winners out of about 200 candidates. At a certain point, a committee member might have already known that he (or she) prefers Obama and Bono to Berlusconi, but might have not decided on the order of Obama and Bono yet. This immediately leads the way to the question whether, given a set of "partial preferences", a certain candidate may still win and hence motivates the study of the Possible Winner problem for k-approval (see Section 7.2 for a formal definition).

In the above described scenarios, the only "unbounded" part of the input is the number of candidates. Hence, directly related questions are whether we can ignore or delete candidates which are not relevant for the decision process and how to identify such candidates. In this context, parameterized algorithmics provides the concept of

kernelization by means of polynomial-time data reduction rules that "preprocess" an instance such that the size of the "reduced" instance only depends on the parameter. Basic definitions are provided in Section 1.3.1. Although kernelization has been applied successfully in many areas (see [34, 126] for surveys), it seems hardly explored for problems in the voting context. In fact, we are only aware of recent results for Dodgson Score [106, 108] (see Chapter 6 for more details) and Swap Bribery [74] as well as some "partial kernelization" results for Kemeny Score, provided in Chapter 4.

In this chapter, we use kernelization to show the fixed-parameter tractability of Possible Winner for k-approval in two "symmetric" scenarios.

1. We consider the combined parameter "number of incomplete votes" t *and* "number of candidates to which every voter gives zero points" $k' := m - k$ for m candidates. Making use of a simple observation we show that Possible Winner admits a polynomial-size problem kernel with respect to (t, k') and provide two algorithms: a simple search tree where the exponential part of the running time is bounded by $2^{O(k')}$ for constant t and a dynamic programming algorithm where the exponential part of the running time is bounded by $2^{O(t)}$ for constant k'. The bound on the dynamic programming table is based the same idea as for the dynamic programming algorithm for Dodgson Score (see Section 6.1). This indicates that this approach may become of general interest.

2. We consider the combined parameter t and k, where k denotes the "number of candidates to which a voter assigns one point". We observe that here one cannot argue symmetrically to the first scenario. Using other arguments, we provide a superexponential-size problem kernel showing the fixed-parameter tractability of Possible Winner with respect to (t, k). For the special case of 2-approval, we give a polynomial-size kernel with $O(t^2)$ candidates by applying an additional reduction rule based on maximum matching techniques. Using a methodology due to Bodlaender et al. [35], our main technical result of this chapter shows that Possible Winner is very unlikely to admit a polynomial-size problem kernel with respect to (t, k).

As in Chapters 7 and 8, all results are given for the unique winner case, that is, looking for a single candidate with maximum score, but they directly transfer to the cowinner case. Note that although the unique-winner and cowinner are used in the definition of Possible Winner and Manipulation in general [65, 131, 194], for k-approval and some of our introductory examples, this seems not to model all situations directly. In particular, using k-approval voting, one often is interested if a distinguished candidate can be part of a winning set of size k. Hence, we discuss other problem variants asking for a set of winners in Chapter 10. However, we stress that in many cases it might also be of interest who is a unique possible winner (the scenario considered in this work). For example, when voting for a board with k members, a unique winner might become the head of the board or get some special award.

Some of the reduction rules given in this chapter will not directly decrease the instance size by removing candidates or votes but instead only decrease the number of possible extensions of a vote, for example, by "fixing" candidates. To *fix* a candidate at a certain position means to specify its relation to all other candidates. Clearly, a candidate may not be fixed at every position in a specific partial vote. For every

candidate $c' \in C$ and a partial vote $v \in V$, let

$$L(v, c') := \{c'' \in C \mid c'' \succ c' \text{ in } v\} \text{ and } R(v, c') := \{c'' \in C \mid c' \succ c'' \text{ in } v\}.$$

Then, fixing a candidate $c' \in C$ *as good as possible* means to add

$$L(v, c') \succ c' \succ C \setminus (L(v, c') \cup \{c'\})$$

to v. Analogously, fixing a candidate *as bad as possible* is realized by adding $C \setminus (R(v, c') \cup \{c'\}) \succ c' \succ R(v, c')$ to v. Fixing a subset of candidates as good/bad as possible means that the single candidates are fixed as good/bad as possible processing them in an arbitrary order. If a candidate $c' \in C$ is fixed in all partial votes, this implies that also its score $s(c')$ is fixed and hence $s(c')$ is well-defined. Furthermore, we say that a candidate c' may shift a candidate c'' to the left (right) in a partial vote v if $c'' \succ c'$ ($c' \succ c''$) in v, that is, setting c' to a one-position (zero-position) implies to set c'' to a one-position (zero-position) as well.

As discussed in the previous chapters, the votes of an input instance of Possible Winner can be partitioned into a (possibly empty) set of linear votes, called V^l, and a set of proper (nonlinear) partial votes, called V^p. We state all our results for the parameter $t := |V^p|$. All positive results also hold for the parameter number of total votes $n := |V^l| + |V^p|$. However, this means that we have to "reduce" the number of linear votes such that it is bounded by the considered parameter. To this end, in some of our reduction rules, we replace the set of linear votes by an *equivalent* set, that is, the maximum partial scores remain unchanged, by using Lemma 7.1 (see Section 7.3.1). To apply Lemma 7.1, for some instances, it might be necessary to add an additional dummy candidate to achieve the Property 1. In all considered cases, this can be done in a straightforward way without changing the parameter values of an instance and thus will not be further discussed. Note that we only state polynomial-size and not provide explicit bounds on the number of linear votes in a reduced instance. This is clearly sufficient to state a polynomial kernel. A further refinement seems not to be of interest from practical point of view since in our case it always make sense to store the maximum partial scores itself instead of "encoding" them into a new set of linear votes of bounded size.

9.1 Fixed number of zero-positions

In this section, we investigate Possible Winner under $(m - k')$-approval with $k' < m$, that is, k' denotes the number of zero-positions. We give a polynomial kernel with respect to (t, k') for Possible Winner where t is the number of partial votes. In addition, we provide two algorithms; a simple branching algorithm with running time $2^{O(k')} \cdot \text{poly}(n, m)$ for constant t and a dynamic programming algorithm with running time $2^{O(t)} \cdot \text{poly}(n, m)$ for constant k'.

9.1.1 Problem kernel

Consider a Possible Winner instance with candidate set C, vote set $V = V^l \cup V^p$, and distinguished candidate $c \in C$ for $(m - k')$-approval. We start with a simple reduction rule that is a crucial first step for all kernelization results in this work.

Rule 9.1. For every vote $v_i \in V^p$ with $|L(v_i, c)| < m - k'$, fix c as good as possible and fix the candidates from $L(v_i, c)$ in a transitivity preserving order.

The condition $|L(v_i, c)| < m - k'$ is crucial since otherwise c might shift a candidate c' to a one-position whereas c is assigned to a zero position and this could cause c' to beat c. The soundness is shown in the following.

Lemma 9.1. *Rule 9.1 is sound and can be carried out in $O(tm^2)$ time.*

Proof. Consider an extension E of an unreduced instance in which c wins. Assume that there is a vote v_i with $|L(v_i, c)| < m - k'$ in which c does not take position $|L(v_i, c)| + 1$ in the extension $E(v_i)$. By definition, for every candidate $c' \in L(v_i, c)$ and for every candidate $c'' \in C \setminus L(v_i, c)$, one cannot have that $c'' \succ c'$. Hence, one can replace $E(v_i)$ by $L(v_i, c) > c > C \setminus (L(v_i, c) \cup \{c\})$ where within $L(v_i, c)$ and within $C \setminus (L(v_i, c) \cup \{c\})$ the candidates have the same order as in $E(v_i)$. Now, distinguish two cases: First, there is a candidate $l \in L(v_i, c)$ that is assigned to a zero-position in $E(v_i)$. Then c must also be assigned to a zero-position in $E(v_i)$. Thus, in the modified extension c must still beat all other candidates since its score is increased by one. Second, all candidates from $L(v_i, c)$ have already taken one-positions in $E(v_i)$. Then, c makes at least as many points as in $E(v_i)$ whereas all other candidates make at most as many points as in $E(v_i)$. Thus, c also wins in the modified extension.

Regarding the running time, for every of the t partial votes, the set $L(v_i, c)$ can be easily computed by checking reachability from c in the following digraph: There is a vertex for every candidate and an arc from c' to c'' if $c'' \succ c'$ in the considered partial vote. This directly leads to the running time $O(tm^2)$. □

After applying Rule 9.1, the score of c is fixed at the maximum possible value since it makes one point in all votes in which this is possible. Now, for every candidate $c' \in C \setminus \{c\}$, by counting the points that c' makes within the linear votes V^l, compute the number of zero positions that c' must assume within the partial votes V^p such that it is beaten by c. Let this number be $z(c')$ and

$$Z_+ := \{c' \in C \setminus \{c\} \mid z(c') > 0\}.$$

Since there are only tk' zero positions in V^p, one can observe the following.

Observation 9.1. *In a yes-instance, it must hold that $\sum_{c' \in C \setminus \{c\}} z(c') \leq tk'$ and $|Z_+| \leq tk'$.*

Observation 9.1 provides a simple upper bound for the number of candidates in Z_+. In the following, we formulate a data reduction rule that bounds the number of remaining candidates, namely,

$$Z_0 := \{c' \in C \setminus \{c\} \mid z(c') = 0\}$$

and replace the linear votes by a bounded number of equivalent votes. The basic idea is that since every remaining candidate from Z_0 can be set arbitrarily in every vote without beating c, it is possible to replace the set of all remaining candidates by tk'^2 "representative candidates".

If $|Z_0| > tk'^2$, we define the following. Let $b_1, \ldots, b_{tk'^2}$ be tk'^2 arbitrary candidates from Z_0. Our goal is to generate an equivalent instance with candidate set $Z_+ \cup \{b_1, \ldots, b_{tk'^2}\} \cup \{c\}$. For a vote $v \in V^p$, let

$$Z_+(v) := \{z \in Z_+ : |R(v, z)| < k'\},$$

that is, $Z_+(v)$ contains the candidates that can take a zero-position within v. For v, define the set of all "relevant" candidates from Z_0 as

$$Z_0(v) := \{z \in Z_0 \mid z \in R(v, z') \text{ for a } z' \in Z_+(v)\}.$$

Since $|Z_+(v)| \leq |Z_+| \leq tk'$ and every candidate from $Z_+(v)$ can shift less than k' other candidates to the right, $|Z_0(v)| < tk'^2$. Hence, the replacement of candidates in the following data reduction rule is well-defined.

Rule 9.2. For every partial vote $v \in V$, let $z_1, \ldots, z_s$ denote the candidates from $Z_0(v)$. If $|Z_0| > tk'^2$, then replace every candidate $z_i \in Z_0(v)$ by the candidate b_i and remove all candidates except $Z_+ \cup \{b_1, \ldots, b_{tk'^2}\} \cup \{c\}$, otherwise Z_0 remains unchanged. Replace the set of linear votes by an equivalent one according to Lemma 7.1.

Making use of Rule 9.2, we arrive at the following.

Theorem 9.1. *For $(m - k')$-approval,* Possible Winner *with t partial votes has a polynomial kernel with at most $tk'^2 + tk' + 1$ candidates and less than $2t^2k'^2 + t^2k'$ votes.*

Proof. To obtain the polynomial kernel, we first apply Rule 9.1 and then Rule 9.2. The soundness of Rule 9.1 has been shown in Lemma 9.1 and the soundness of Rule 9.2 is easy to see: Every extension of an unreduced instanced can be transferred to an extension for the reduced instance by replacing the candidates from Z_+ as described in Rule 9.2. The computation of $Z_0(v)$ can be accomplished in $O(tm^2)$ time using the reachability graph described in the proof of Lemma 9.1 and the replacement of the linear votes can be done in polynomial time according to Lemma 7.1. □

9.1.2 Parameterized algorithms

Applying any brute-force algorithm after the kernelization directly leads to an fixed-parameter algorithm. For example, one might guess which candidates assume the last k' position for every of the t votes and check if this results in a winning extension. Systematically checking all such "guesses" for a reduced instance with $tk'^2 + tk' + 1$ candidates leads to an overall running time of $\binom{tk'^2+tk'+1}{k'} \cdot \text{poly}(n, m)$. We now investigate more efficient strategies. Herein, we are especially interested in algorithms with an exponential running time factor of the form $2^{O(p)}$ where p denotes either k' or t while the other parameter is of constant value. The brute-force algorithm does not imply such a running time for k' or t. In the following, we describe such algorithms for the special cases $t = 2$ and $k' = 2$ and explain how to extend them to greater constant values.

Initialization:
For every $D' \in \mathcal{D} \setminus \{(d_1, \ldots, d_p)\}$, set $T(0, D') = 0$.
Set $T(0, (d_1, \ldots, d_p)) = 1$.

Update:
For $0 \leq i \leq t-1$,
for every $D' = (d'_1, \ldots, d'_p) \in \mathcal{D}$,
$T(i+1, D') = 1$ if there are two candidates z_g, z_h that can take the zero-positions in v_{i+1}
and $T(i, D'') = 1$ with $D'' := \{d''_1, \ldots, d''_p\}$ and
$d''_j = d'_j$ for $j \in \{1, \ldots, q\} \setminus \{g, h\}$, $d''_g \leq d'_g + 1$, and $d''_h \leq d'_h + 1$.

Output:
"yes" if $T(t, (0, \ldots, 0)) = 1$, "no" otherwise

Figure 9.1: Dynamic programming algorithm for $(m-2)$-approval.

Constant number of partial votes. Consider a Possible Winner instance after applying Rule 9.1, that is, the distinguished candidate is fixed in all votes. For two partial votes, according to Observation 9.1 there can be at most $2k'$ candidates that must take a zero-position in a yes-instance. Every such candidate must take a zero-position in the first or in the second vote. Hence, one can branch into these two possibilities for every candidate, and then check if there is a corresponding extension. This results in a search tree of size at most $2^{2k'} = 4^{k'}$. For every vote, checking whether there exists a corresponding extension can be accomplished in $O(m)$ time. More specifically, if there is a linear extension, it can be found in $O(m)$ time by topological sorting [66].

Now, consider the case of having a constant number t of votes. For every candidate that has to be assigned to s zero-positions with $0 < s < t$ in a winning extension, there are

$$\binom{t}{s} \leq \binom{t}{t/2} < 2^t$$

possibilities of choosing the zero-positions within the t partial votes. Since there are at most tk' such candidates (Observation 9.1), this yields a search tree of size less than $(2^t)^{tk'}$. Hence, we arrive at the following.

Proposition 9.1. *For t partial votes,* Possible Winner *for $(m-k')$-approval can be solved in $O(2^{t^2k'} \cdot nm + nm^2)$ time.*

For constant t, Proposition 9.1 directly leads to the exponential running-time factor $2^{O(k')}$.

Constant number of zero-positions. For constant k' the existence of an algorithm with running time $2^{O(t)} \cdot \mathrm{poly}(n, m)$ seems to be less obvious than for the case of constant t. We start by giving a dynamic programming algorithm for $(m-2)$-approval running in $4^t \cdot O(nm^2)$ time and space. For the analysis, we employ the same idea as for the dynamic programming algorithm for Dodgson Score (Figure 6.4).

As in the previous subsection, fix c according to Rule 9.1 such that it makes the maximum possible score and let $Z_+ := \{z_1, \ldots, z_p\}$ denote the set of candidates that

take at least one zero-position in a winning extension. Let $d_1, \ldots, d_p$ denote the corresponding number of zero-positions that must be assumed and let

$$\mathcal{D} := \{(d'_1, \ldots, d'_p) \mid 0 \leq d'_j \leq d_j \text{ for } 0 \leq j \leq p\}.$$

The dynamic programming table T is defined by $T(i, D')$ for $1 \leq i \leq t$ and $D' = (d'_1, \ldots, d'_p) \in \mathcal{D}$. Herein, $T(i, D') = 1$ if the partial votes from $\{v_1, \ldots, v_i\}$ can be extended such that candidate z_j takes at least $d_j - d'_j$ zero-positions for every $j \in \{1, \ldots, p\}$; otherwise $T(i, D') = 0$. Intuitively, d'_j stands for the number of zero-positions that z_j must still take in the remaining votes $\{v_{i+1}, \ldots, v_t\}$. If $T(t, (0, \ldots, 0)) = 1$ for an instance, then it is a yes-instance. The dynamic programming algorithm is stated in Figure 9.1 and leads to the following.

Theorem 9.2. *For $(m-2)$-approval with t partial votes,* Possible Winner *can be solved in $O(4^t \cdot nm^2)$ time and $O(4^t \cdot t)$ space.*

Proof. We first show the correctness of the dynamic programming algorithm given in Figure 9.1. Regarding the initialization, within the empty set of votes, that is, for $i = 0$, no candidate can be assigned to any zero-position and hence $T(0, D')$ can only be true ("1") if $D' = (d_1, \ldots, d_p)$. Regarding the update step, $T(i+1, D') = 1$ means that within $\{v_1, \ldots, v_{i+1}\}$, it must be possible that every candidate $z_j \in Z_+$ is assigned to $d_j - d'_j$ zero-positions. Since there are only two zero-positions in v_{i+1}, all but two candidates, z_g and z_h, must be assigned to the required number of zero-positions in $\{v_1, \ldots, v_i\}$. Hence, there must be an entry $T(i, D'')$ with $T(i, D'') = 1$ and $d''_j = d'_j$ for all $j \in \{1, \ldots, p\} \setminus \{g, h\}$, $d''_g \leq d'_g + 1$, and $d''_h \leq d'_h + 1$ which equals the conditions given by the dynamic programming algorithm. Note that relaxing the condition "$d''_z = d'_z + 1$" to "$d''_z \leq d'_z + 1$" has the effect that a candidate can take more zero-positions than required without "bookkeeping" this by negative values of the entries from D'. Finally, a considered instance is a yes-instance if and only if within all partial votes every candidate $z_j \in Z_+$ takes at least d_j zero-position, that is, the corresponding table entry $T(t, (0, \ldots, 0))$ must be true.

Now, to analyze the size of the dynamic programming table, we make use of two conditions: first, $|Z_+| \leq 2t$, and, second, $\sum_{i=1}^{|Z_+|} d_i \leq 2t$ (see Observation 9.1). With these conditions, it is easy to verify that $|\mathcal{D}| = \prod_{i=1}^{|Z_+|}(d_i + 1) \leq 2^{2t} = 4^t$, resulting in a total table size of $O(t \cdot 4^t)$. Clearly, the initialization and the update step can be carried out in polynomial time per table entry and the running time bound follows. □

It is not hard to see that the given algorithm can be extended to work for other constant values of k' using the same definition of the dynamic programming table. Since $|Z_+| \leq tk'$ and $\sum_{i=1}^{|Z_+|} d_i \leq tk'$ (Observation 9.1), the table size is bounded by $t \cdot 2^{tk'}$. We only need to slightly modify the update step. Here, one needs to consider candidate subsets of size k' instead of pairs of candidates that can take the zero-position in the "current" vote. Since there are at most $\binom{tk'}{k'} < (tk')^{k'}$ such subsets, this gives an additional factor of less than $(tk')^{k'}$ to the running time which is still polynomial in t for constant k'. Hence, altogether, one arrives at the following.

Theorem 9.3. *For $(m-k')$-approval,* Possible Winner *with t partial votes can be solved in $2^{O(t)} \cdot O(nm^2)$ time for constant k'.*

Proposition 9.1 and Theorem 9.3 also provide the following running time bound with respect to the combined parameter (t, k').

Corollary 9.1. *For $(m-k')$-approval,* Possible Winner *with t partial votes can be solved in $O(\min\{2^{t^2k'},\ 2^{tk'} \cdot (tk')^{k'}\} \cdot nm^2)$ time.*

9.2 Fixed number of one-positions

In this section, we study Possible Winner for k-approval with respect to the combined parameter k and "number of partial votes" t. The problem can be considered as "filling" tk one-positions such that every candidate is still beaten by c. In the previous section, we exploited that the number of candidates that must take a zero-position is already bounded by the combined parameter t and "number of zero-positions" in a yes-instance (Observation 1). Here, we cannot argue analogously: Our combined parameter (t, k) only bounds the total number of one-positions but there can be an unbounded number of candidates that may take a one-position in different winning extensions of the partial votes. Hence, we argue that if there are too many candidates that can take a one-position, then there must be several choices that lead to a valid extension. We show that it is sufficient to keep a set of "representative candidates" that can take the required one-positions if and only if this is possible for the whole set of candidates. This results in a problem kernel of super-exponential size showing fixed-parameter tractability with respect to (t, k). We complement this result by showing that it is very unlikely that there is a kernel of polynomial size. For 2-approval, from the reduction rules used to show the super-exponential kernel for general k, one directly obtains a polynomial kernel with $O(t^3)$ candidates. Using an additional reduction rule based on a structural property of maximum matchings in bipartite graphs, we improve this to a polynomial kernel with $O(t^2)$ candidates.

9.2.1 Problem kernels

We first describe a kernelization approach for Possible Winner for k-approval in general and then show how to obtain a better bound on the kernel size for 2-approval.

Problem kernel for k-approval.

In order to describe more complicated reduction rules, we assume that a considered instance is exhaustively reduced with respect to some simple rules. To this end, we fix the distinguished candidate c as good as possible by applying Rule 9.1 (using that $m - k' = k$). Afterward, we apply a simple reduction rule to get rid of "irrelevant" candidates and to check whether an instance is a trivial no-instance:

Rule 9.3.

1. For every candidate $c' \in C \setminus \{c\}$, if making one point in the partial votes causes c' not to be beaten by c, then fix c' as bad as possible in every vote.
2. Compute a set D of candidates that can be deleted: For every candidate $c' \in C \setminus \{c\}$ with $|L(v, c')| > k$ for all $v \in V^p$, if the score $s(c')$ is at least $s(c)$, then output "no solution", otherwise add c' to D. Delete D from V^p and replace V^l by an equivalent set for the candidates from $C \setminus D$ using Lemma 7.1.

$$v: \quad a \succ b \succ d \succ x, \quad e \succ f, \quad g \succ f \succ y \succ c, \quad a \succ h$$

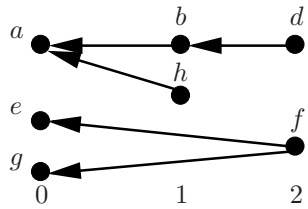


Figure 9.2: Example for 3-approval: Partial vote v (left-hand side) and corresponding digraph with levels 0, 1, and 2. Arcs following by transitivity are omitted. Note that x, y, and c do not appear in the digraph since they are irrelevant for v.

The soundness of Rule 9.3 is not hard to see: Every candidate fixed by the first part cannot be assigned to a one-position in any winning extension. For the second part, every winning extension of an unreduced instance can easily be transformed into a winning extension for the reduced one by deleting the candidates specified by Rule 9.3 and *vice versa*.

In the following, we assume that Rule 9.3 has been applied, that is, all remaining candidates can make at least one point in a winning extension. To state further reduction rules, a partial vote v is represented as a digraph with vertex set

$$\{c' \mid c' \in C \setminus \{c\} \text{ and } |L(v, c')| < k\}.$$

All other candidates are considered as *irrelevant* for this vote since they cannot take a one-position. The vertices are organized into k levels from 0 to $k-1$. More precisely, for $j \in \{0, \ldots, k-1\}$, let

$$L_j(v) := \{c' \mid c' \in C \setminus \{c\} \text{ and } |L(v, c')| = j\},$$

that is, $L_j(v)$ contains all candidates that shift exactly j candidates to a one-position if they are assigned to the best possible position. There is a directed arc from c' to c'' if and only if $c'' \in L(v, c')$. Figure 9.2 displays an example for the representation of a partial vote for 3-approval.

Without data reduction, the number of candidates per level may be arbitrary large. For some cases it is easy to see that one can "delete" all but some representative candidates. The following reduction rule provides such an example for the case that there are at least tk candidates that can take the first position.

Rule 9.4. For $v \in V^p$ with $|L_0(v)| \geq tk$, consider an arbitrary subset $L' \subseteq L_0(v)$ with $|L'| = tk$. Add $L' \succ C \setminus L'$ to v.

The soundness of Rule 9.4 can be seen as follows. Consider a winning extension E for a nonreduced instance and a vote $v \in V^p$ with $|L_0(v)| \geq tk$. Since there are tk one-positions in the partial votes, there must be at least k candidates from L' not having assumed a one-position within the other $t-1$ votes. Setting these k candidates to the one-positions in v leads to a winning extension of the reduced instance. The other direction is obvious. Note that Rule 9.4 explores the fact that a candidate from $L_0(v)$ can be set to an arbitrary one-position without shifting any other candidate. This does not hold for any other level.

If Rule 9.4 applies to all partial votes, then in a reduced instance at most t^2k candidates are not fixed at zero positions in all partial votes and the remaining candidates can be deleted by Rule 9.3. Hence, we consider the situation that there is a partial vote v with $|L_0(v)| < tk$. In this case, we cannot ignore the candidates from the other levels but replace them by a bounded number of representatives. We first discuss how to find a set of representatives for 2-approval and then extend the underlying idea to work for general k.

For 2-approval, for a vote v with $|L_0(v)| < 2t$, it remains to bound the size of $L_1(v)$. This is achieved by the following reduction rule:

Rule 9.5. Fix all but $2t$ in-neighbors of every candidate from $L_0(v)$ at zero-positions.

Given a winning extension E for the nonreduced instance, a winning extension E' for v in the reduced instance can be obtained as follows. In $E(v)$ the first position must be assigned to a candidate c' from $L_0(v)$ and c' can also be assigned to the first position in $E'(v)$. If there is another candidate from $L_0(v)$ taking the second position in $E(v)$, then one can do the same in $E'(v)$. Otherwise, distinguish two cases.

1. c' has less than $2t$ in-neighbors, then the reduction rule has not fixed any candidate that shifts c' to the first position and thus v can be extended in the same way as in E.

2. c' has at least $2t$ in-neighbors. Since there are only $2t$ one-positions corresponding to the partial votes and $2t$ nonfixed in-neighbors, the second position of v can be assigned to a candidate that does not take a one-position in any other vote of E.

Altogether, for 2-approval, one ends up with less than $4t^2$ nonfixed candidates per vote and hence with $O(t^3)$ nonreduced candidates in total. For general k, we extend this approach iteratively by bounding the number of candidates for every level of a partial vote. To this end, we give the following reduction rule which clearly subsumes Rule 9.5.

Rule 9.6. Consider a partial vote $v \in V^p$ with $|L_0(v)| < tk$. Start with $i = 1$ and repeat until $i = k$.
- For every candidate $c' \in L_i(v)$, if there are more than tk candidates in $L_i(v)$ which have the same neighborhood as c' in $L_0(v) \cup L_1(v) \cup \cdots \cup L_{i-1}(v)$, fix all but tk of them as bad as possible.
- Set $i := i + 1$.

Note that if the distinguished candidate c is in $L_i(v)$ for any i, due to Rule 9.1 one has $|L_j(v)| = 1$ for all $j \leq i$. Hence c cannot be contained in $L_j(v)$ with $|L_j(v)| \geq tk$ and will never be fixed at a zero-position by Rule 9.6.

Lemma 9.2. *Rule 9.6 is sound and can be carried out in $O(m^2 \cdot k)$ time.*

Proof. Every winning extension of an instance reduced by Rule 9.6 is clearly also a winning extension of the unreduced instance. We describe how to obtain a winning extension for the reduced instance from a winning extension E of the unreduced instance. For $E(v)$, let r denote the leftmost candidate which is fixed as bad as possible

in v by Rule 9.6. Let $\text{pos}(r)$ denote the position from r in $E(v)$. If a candidate r does not exist or assumes a zero-position in $E(v)$, one can assign the one-positions of v in the reduced instance to the same candidates as in $E(v)$. Otherwise, since r has been fixed by Rule 9.6, there must be tk nonfixed candidates shifting exactly the same candidates to the left as r, namely

$$R := \{c' \in C \setminus \{c, r\} \mid L(v, r) = L(v, c')\}.$$

Since in $V^p \setminus \{v\}$ there are only $(t-1)k$ one-positions, one can assign the positions ranging from $\text{pos}(r)$ to k to candidates from R which do not assume a one-position in any other vote from E and assign the zero-positions to the remaining candidates in a transitivity preserving order. Now, the only candidates that can make more points in the constructed extension than in E are the candidates occurring in R. More precisely, a candidate from R can make one point in the extension of v if it makes zero points in $V^p \setminus \{v\}$. Hence, one obtains a winning extension for the reduced instance.

Regarding the running time, for each of the k levels one must check the neighborhood of every candidate in this level. Using some appropriate data structure, one can classify the at most m candidates per level by iterating once over the neighborhood ($\leq m$) of every candidate. □

Theorem 9.4. *For k-approval,* Possible Winner *admits a problem kernel with size bounded by a computable function in k and the "number of partial votes" t.*

Proof. The proof combines several reduction rules whose soundness and polynomial running time has been discussed above. First, apply Rule 9.1 and Rule 9.3 to fix c and to delete irrelevant candidates. Second, apply Rule 9.4 exhaustively. As a consequence, it holds that $|L_0(v)| \leq tk$ for every vote $v \in V^p$. Third, for every vote $v \in V^p$ with $|L_0(v)| < tk$, build up the level digraph and apply Rule 9.6. Now, we argue that after applying Rule 9.6 to a partial vote v, the number of nonfixed candidates in v is bounded. More specifically, we show that for every "iteration step" of Rule 9.6 the number of candidates that remain in the considered level is bounded by a function depending only on the number of candidates in the previous levels, k, and t. Since $|L_0| < tk$ and one runs through at most k iteration loops, this gives an upper bound on the number of nonfixed candidates per vote. For $1 \leq i \leq k-1$, let f_{i-1} denote the number of candidates in $\bigcup_{0 \leq j \leq i-1} L_j$. Then, in $L_i(v)$, there can be at most $\binom{f_{i-1}}{i}$ candidates with different out-neighborhoods. For every such out-neighborhood, one keeps at most tk candidates. It follows that the number of "nonfixed" candidates is bounded in every vote. Since all candidates that are fixed in all votes can be removed by Rule 9.3 and the linear votes can be replaced according to Lemma 7.1 by a bounded number of linear votes, the theorem follows. □

It directly follows from Theorem 9.4 that Possible Winner for k-approval is fixed-parameter tractable with respect to the combined parameter (t, k). Although the given analysis of the kernelization is only of theoretical interest, the described reduction rules might still be useful in practice. Clearly, it is desirable to improve the size of the problem kernel, for example, by using termination conditions within the iteration loops and/or "global" reduction rules that consider more than one vote at a time. In the following, we employ such a global reduction rule to obtain an improved bound of the problem kernel size for the important special case of 2-approval voting.

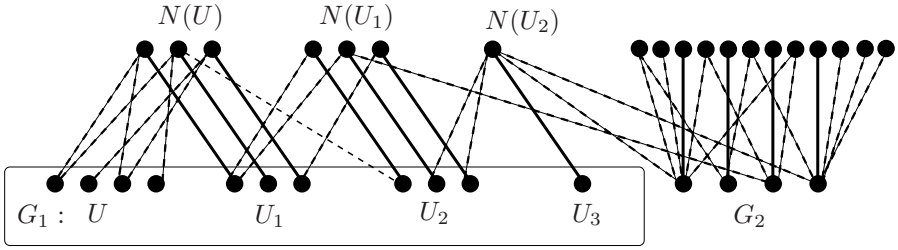


Figure 9.3: Illustration of Lemma 9.3. Matching edges are solid and the remaining edges dashed. The subset G_1 consists of $U \cup U_1 \cup U_2 \cup U_3$ where U_i denotes the content assigned to U in the ith iteration.

Improved problem kernel for 2-approval.

As discussed above, the kernelization as stated for k-approval in general leads to a polynomial kernel with $O(t^3)$ candidates for 2-approval. We make use of the following lemma to arrive at a polynomial kernel with $O(t^2)$ candidates. Note that similar arguments are used in several works, see [55, 171]. Hence, our contribution is mainly to come up with an appropriate modeling allowing for the application of the following lemma to a voting problem.

Lemma 9.3. *For a bipartite graph $(G \cup H, F)$ with maximum matching M, there is a partition of G into $G_1 \uplus G_2$, such that the following holds. First, all neighbors of G_1 are part of M. Second, every vertex from G_2 has a matching neighbor outside $N(G_1)$.*

Since we are not aware of an explicit statement of the form as given by Lemma 9.3, we provide a proof.

Proof. Construct a partition of G as follows (see Figure 9.3 for an example). Start with empty vertex set G_1. Let U denote the set of unmatched vertices in G. Repeat until $N(U) \setminus (N(G_1) \cap N(U))$ is empty (that is, until G_1 is remains unchanged):

- Add U to G_1.
- Let U' be the set containing the matching neighbors from $N(U)$.
- Set $U := U'$.

Finally, add U to G_1. The set G_2 consists of all remaining vertices from G, that is, $G_2 := G \setminus G_1$. We show by contradiction that all neighbors of G_1 must be matching vertices. Let U_i denote the set of vertices assigned to U in the ith iteration of the repeat loop and assume that there is a vertex $u_i \in U_i$ which has a nonmatching neighbor h. By construction, there must be a path from a vertex $u \in U$ to u_i such that every second edge is in M. Clearly, this path has length $2i$ and contains i matching edges. Changing M by replacing all matching edges in the considered path by the nonmatching edges from this path and by adding the edge between u_i and its unmatched neighbor h results in a matching of size $|M| + 1$, a contradiction.

By construction there cannot be a matching edge from a vertex from $N(G_1)$ to a vertex in G_2 and every vertex in G_2 must have a matching neighbor. Hence, the second part of the lemma follows. □

In the following, we assume that Rule 9.1 and Rule 9.3 have been applied. To state our new reduction rule, we define a bipartite graph $(G \cup H, F)$ as follows. For a partial profile with partial votes V^p, let $V' := \{v' \in V^p : |L_0(v')| < 2t\}$. For every $v'_i \in V'$, for $1 \leq j \leq |L_0(v'_i)|$, add a vertex g_i^j to G. Intuitively, for every candidate that can take a first position in v'_i there is a corresponding vertex in G. If a candidate can take the first position in several votes, then there are several vertices corresponding to this candidate. The vertex set H contains one vertex for every candidate from

$$(\bigcup_{v' \in V'} L_1(v')) \setminus (\bigcup_{v' \in V'} L_0(v')).$$

There is an edge between $g_i^j \in G$ and $h \in H$ if setting the candidate corresponding to h to the second position in v'_i shifts the candidate corresponding to g_i^j to the first position. Now, we can state the following.

Rule 9.7. Compute a maximum matching M in $(G \cup H, F)$. Fix the candidates corresponding to the nonmatched vertices in H as bad as possible in every vote from V'.

Lemma 9.4. *Rule 9.7 is sound and can be carried out in $O(nm^2 + mt^3)$ time.*

Proof. A winning extension for an instance reduced with respect to Rule 9.7 is also a winning extension for an unreduced instance. Now, we show the other direction. Given a winning extension E for an unreduced instance, we construct a winning extension E_r for a reduced instance. Since Rule 9.7 does not fix any candidate which can take the first position in at least one vote, the first positions in E_r can be assumed by the same candidates as in E. It remains to fix the second positions without beating c. For every vote v_i, let g_i^e denote the candidate that takes the first position in v_i in E. For the corresponding vertex g_i^e, distinguish two cases:

1. $g_i^e \in G_1$. In this case, none of the neighbors of g_i^e have been fixed and, thus, the candidate which takes the second position in v_i in E can also take the second position E_r.

2. $g_i^e \in G_2$. In this case, assign the candidate corresponding to the matching neighbor from g_i^e to the second position.

Now, it is not to hard to see that c wins in E_r: The only candidates that possibly make more points in E_r than in E are the candidates corresponding to the matching neighbors of vertices from G_2. Due to the matching property, every such candidate makes one point in at most one vote from V'. By definition, G only contains vertices such that the corresponding candidates can make at least one point. For all votes from $V^p \setminus V'$ one can easily find a winning extension which does not assign the "matching-candidates" to one-positions (see Rule 9.3). It follows that c also wins in the extension E_r.

To show the running time, we first consider the size of the constructed bidirected graph $(G \cup H, F)$. The set G contains for every partial vote v a vertex for every

candidate from $L_0(v)$ if $|L_0(v)| < 2t$. Hence, $|G| < 2t^2$. An upper bound for H is provided by the total number of candidates m. Since every candidate from H can shift at most one candidate per partial vote, one obtains $|F| < mt$. Clearly, $(G \cup H, F)$ can be build in $O(nm^2)$ time and since a maximum bipartite matching can be found in $O(|F| \cdot |G \cup H|)$ time [66] the claimed running time follows. □

Bounding the size of candidates in level 0 by Rule 9.4 and the (remaining) candidates in level 1 by Rule 9.7 one arrives at the following.

Theorem 9.5. *For 2-approval with t partial votes,* Possible Winner *admits a polynomial kernel with at most $4t^2$ candidates.*

Proof. The kernelization algorithm combines different reduction rules whose soundness and polynomial running time is either obvious or has been shown before (Lemma 9.4). As a first step of data reduction apply Rule 9.1 and Rule 9.3. Then, c is fixed and all remaining nonfixed candidates can make at least one point in the remaining votes. Now, by applying Rule 9.4 one achieves that $|L_0| \leq 2t$ for every vote and thus $|\bigcup_{v \in V} L_0(v)| \leq 2t^2$. Next, apply Rule 9.7. Then, the total number of nonfixed candidates from $\bigcup_{v \in V} L_1(v)$ equals the size of a maximum matching in the bipartite auxiliary graph which can be at most $|G|$. Since for every vote one adds less than $2t$ candidates to G, one obtains an upper bound of $2t^2$ for $|\bigcup_{v \in V} L_1(v)|$ as well. Hence, by deleting all candidates that are fixed at zero-positions in all votes by Rule 9.3 (except the distinguished candidate c), one ends up with an instance of at most $4t^2$ candidates. Finally, the linear votes can be replaced according to Lemma 7.1. □

9.2.2 Kernel lower bound

In the previous subsection, we provided a kernel of super-exponential size with respect to (t, k) for Possible Winner under k-approval. Here, we complement this result by showing that for k-approval, Possible Winner cannot have a polynomial kernel with respect to (t, k) under some reasonable assumptions from classical complexity theory. To this end, we apply a method introduced by Bodlaender et al. [35] and Fortnow and Santhanam [114] which is briefly described in the following.

Definition 9.1. [35] A *composition algorithm* for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm that receives as input a sequence $((x_1, p), \ldots, (x_q, p))$ with $(x_i, p) \in \Sigma^* \times \mathbb{N}$ for each $1 \leq i \leq q$, uses time polynomial in $\sum_{i=1}^{q} |x_i| + p$, and outputs $(y, p') \in \Sigma^* \times \mathbb{N}$ with

- $(y, p') \in L \Leftrightarrow (x_i, p) \in L$ for some $1 \leq i \leq q$ and
- p' is polynomial in p.

A parameterized problem is *compositional* if there is a composition algorithm for it. Note that this definition directly extends to parameters that are constant-size tuples of integers. For a parameterized problem L, the *unparameterized version* L^u is the language $\{x\#1^k \mid (x, k) \in L\}$ where 1 is an arbitrary fixed letter in Σ and $\# \notin \Sigma$.

Theorem 9.6. *[35, 114] Let L be a compositional parameterized problem whose unparameterized version is NP-complete. Then, unless* coNP $\subseteq$ NP / poly*, there is no polynomial kernel for L.*

The assumption that coNP $\not\subseteq$ NP / poly is supported by a series of papers showing the collapse of the polynomial hierarchy to different levels: Karp and Lipton [141] showed that coNP $\subseteq$ NP / poly implies that the polynomial hierarchy collapses to the third level. This has been strengthed to an collapse to $\mathrm{ZPP}^{\mathrm{NP}^{\mathrm{NP}}}$ [151]. A further substantial improvement has been provided by Cai et al. [49] showing that coNP $\subseteq$ NP / poly already implies a collapse to $\mathrm{S}_2^{\mathrm{NP}}$.

For Possible Winner parameterized with respect to (t, k), it is easy to see that the unparameterized version is NP-complete as well. Hence, the main work to apply Theorem 9.6 is to achieve a composition algorithm. Composition algorithms have been provided for several fundamental combinatorial problems, see for example [36, 73]. In particular, Dom et al. [73] introduced a general framework to build composition algorithms employing so-called "identifiers". One of the necessary conditions to apply this framework, is the existence of an algorithm running in $2^{p^\gamma} \cdot \mathrm{poly}$ time for the considered parameter p and a fixed constant γ. Considering the combined parameter "number of ones" k and "number of partial votes" t for Possible Winner under k-approval, there is no known algorithm running in $2^{(tk)^\gamma} \cdot \mathrm{poly}$ time. Hence, we apply the following overall strategy (which might be also useful for other problems).

Overall strategy. We employ a proof by contradiction. Assume that there is a polynomial kernel with respect to (t, k). Then, since for Possible Winner there is an obvious brute-force algorithm running in $m^{tk} \cdot \mathrm{poly}(n, m)$ time for m candidates and n votes, there must be an **Algorithm S** with running time $\mathrm{poly}(t, k)^{tk} \cdot \mathrm{poly}(n, m) < 2^{(tk)^\gamma} \cdot \mathrm{poly}(n, m)$ for an appropriate constant γ. In the next paragraph, we use the existence of Algorithm S to design a composition algorithm for the combined parameter (t, k). Since it is easy to verify that the unparameterized version of Possible Winner is NP-complete, it follows from Theorem 9.6 that unless coNP $\subseteq$ NP / poly there is no polynomial kernel with respect to (t, k), a contradiction under the assumption that coNP $\not\subseteq$ NP / poly. Altogether, it remains to give a composition algorithm.

Composition algorithm. Consider a sequence $((x_1, (t, k)), \ldots, (x_q, (t, k)))$ of q Possible Winner instances for k-approval. To simplify the construction, we make two assumptions. First, we assume that there is no "obvious no-instance", that is, an instance in which a candidate c' is not beaten by c even if c' makes zero points in all of the partial votes. This does not constitute any restriction since such instances can be found and removed in time polynomial in $\sum_{i=1}^{q} |x_i|$. Second, we assume that for $x_j, 1 \leq j \leq q$, within the partial votes the distinguished candidate makes zero points in every extension. Since it follows from the construction used in Lemma 7.3 that the unparameterized version of the problem remains NP-complete for this case, this assumption leads to a nonexistence result for this special case and thus also for the general case.

The overall structure of the composition algorithm is described as follows. If $q > 2^{(tk)^\gamma}$ for γ as specified for Algorithm S, the composition algorithm applies S to every instance. This can be done within the running time bound required by Definition 9.1. Hence, in the following, we assume that the number of instances is at most $2^{(tk)^\gamma}$. As suggested by Dom et al. [73], this can be used to assign an "identifier" of sufficiently small size to every instance. Basically, the identifiers, which will be realized by specific

sets of candidates, rely on the binary representation of the numbers from $\{1, \ldots, q\}$. The size of an identifier will be linear in $s := \lceil \log q \rceil$ which is polynomial in the combined parameter (t, k) since $q \leq 2^{(tk)^\gamma}$.

Now, we provide a composition algorithm for the case that $q \leq 2^{(tk)^\gamma}$. Compose the sequence of instances to one big instance

$$(X, (3s + 4, 2t)) \text{ with } X = (C, V^l \cup V^p, c)$$

as follows. For $1 \leq i \leq q$, let x_i be $(C_i, V_i^l \cup V_i^p, c_i)$. Then,

$$C := \biguplus_{1 \leq i \leq q} (C_i \setminus \{c_i\}) \uplus \{c\} \uplus D \uplus Z \uplus A \uplus B$$

with

- $D := \{d_0^0, \ldots, d_s^0\} \cup \{d_0^1, \ldots, d_s^1\}$,
- $Z := \bigcup_{1 \leq j \leq t} Z_j$ with $Z_j := \{z_{h,j}^0 \mid 0 \leq h \leq s\} \cup \{z_{h,j}^1 \mid 0 \leq h \leq s\}$,
- $A := \{a_1, \ldots, a_q\}$,
- and a set B with $|B| := 2s + 3 - k$.

The candidates from D and Z will be used as identifiers for the different instances. More specifically, every instance x_i is uniquely identified by the binary code of the integer $i = b_0 \cdot 2^0 + b_1 \cdot 2^1 + \cdots + b_s \cdot 2^s$ with $b_h \in \{0, 1\}$ leading to the following definition.

Definition 9.2. A subset $D_i \subset D$ *identifies* x_i when $d_h^1 \in D_i$ if and only if $b_h = 1$ and $d_h^0 \in D_i$ if and only if $b_h = 0$.

Let $\overline{D_i} := D \setminus D_i$. Similarly, for every $1 \leq j \leq t$, the set $Z_{i,j}$ denotes the candidates from Z_j that identify i, that is,

$$Z_{i,j} := \{z_{h,j}^0 \mid h \in \{0, \ldots, s\} \text{ and } b_h = 0\} \cup \{z_{h,j}^1 \mid h \in \{0, \ldots, s\} \text{ and } b_h = 1\}.$$

Let $\overline{Z_{i,j}} := Z_j \setminus Z_{i,j}$ denote the remaining candidates from Z_j.

The set of partial votes V^p consists of two subsets V_1^p and V_2^p, both containing t partial votes. The basic idea is that a winning extension of V_1^p "selects" an instance x_i and there is a winning extension for x_i if and only if V_2^p can be extended such that c wins. The set V_1^p contains the vote

$$\{Z_{i,1} \cup \overline{D_i} \cup \overline{Z_{i,t}} \succ a_i \mid 1 \leq i \leq q\},\ D \cup Z \cup A \succ C \setminus (D \cup Z \cup A),$$

meaning that the vote contains the constraints $Z_{i,1} \cup \overline{D_i} \cup \overline{Z_{i,t}} \succ a_i$ for every i. Furthermore, for every $j \in \{2, \ldots, t\}$, the set V_1^p contains the vote

$$\{Z_{i,j} \cup \overline{D_i} \cup \overline{Z_{i,j-1}} \succ a_i \mid 1 \leq i \leq q\},\ D \cup Z \cup A \succ C \setminus (D \cup Z \cup A).$$

The set V_2^p consists of the partial votes $v_1, \ldots, v_t$. Every vote $v_j \in V_2^p$ "composes" the votes v_i^j for $i \in \{1, \ldots, q\}$ where v_i^j denotes the jth vote from instance x_i after deleting c_i. Then, for $j \in \{1, \ldots, t\}$, the vote v_j is

$$B \succ (C \setminus B),\ \{v_i^j \mid 1 \leq i \leq q\},\ \{D_i \succ C_i \setminus \{c_i\} \mid 1 \leq i \leq q\},\ C \setminus (A \cup Z \cup \{c\}) \succ A \cup Z \cup \{c\}.$$

Using Lemma 7.1, we construct a set V_l of linear votes polynomial in $|C|$ and $|V^p|$ such that the following hold.

V_1^p :	$Z_{w,1}$	$> \overline{D_w}$	$> \overline{Z_{w,t}}$	$> a_w$	$> C \setminus (Z_{w,1} \cup \overline{D_w} \cup \overline{Z_{w,t}})$	
	$Z_{w,j}$	$> \overline{D_w}$	$> \overline{Z_{w,j-1}}$	$> a_w$	$> C \setminus (Z_{w,j} \cup \overline{D_w} \cup \overline{Z_{w,j-1}})$	for $2 \leq j \leq t$
V_2^p :	B	$> D_w$	$> w_j$		$> C \setminus (B \cup D_w \cup (C_w \setminus \{c_w\}))$	for $1 \leq j \leq t$

Figure 9.4: Extension for X in which c wins. For a winning extension $E(x_w) = w'_1, \ldots, w'_t$ of x_w, let w_j denotes the linear order given by w'_j restricted to the candidates from $C_w \setminus \{c\}$. The remaining subsets of candidates are fixed in any transitivity preserving order.

- For $i \in \{1, \ldots, q\}$, the maximum partial score of every candidate $c' \in C_i \setminus \{c_i\}$ equals the maximum partial score of c' in x_i.
- For every candidate from $A \cup D \cup B$, the maximum partial score is t.
- For every candidate from Z, the maximum partial score is one.

Lemma 9.5. *The constructed instanced X is a yes-instance for $(3s+4)$-approval if and only if there is an $i \in \{1, \ldots, q\}$ such that x_i is a yes-instance for k-approval.*

Proof. "$\Leftarrow$": Assume there is an instance x_w for which c is a possible winner. Let $E(x_w) = w_1, \ldots, w_t$ denote a winning extension for x_w and recall that C_w denotes the set of candidates from x_w. Then, extend the partial votes from X as indicated in Figure 9.4. Since there are $3s+4$ one-positions per vote, $|D_i| = s+1$, and $|B| = 2s+3-k$, in every extended vote from V_2^p, there are k one-positions that are assumed by candidates from $C_w \setminus \{c_w\}$. Because of this and due to the equivalence of the partial orders in the corresponding votes, the candidates from $C_w \setminus \{c_w\}$ make exactly the same number of points in the extension for X as in $E(x_w)$ and are beaten by c. The remaining "instance candidates", namely, $\bigcup_{i \neq w} C_i \setminus \{c_i\}$ do not make any points in the given extension and thus are beaten by c. The candidates from D can be partitioned into the two disjoint subsets D_w and $\overline{D_w}$. The candidates from D_w make t points in V_2^p and zero points in V_1^p whereas the candidates from $\overline{D_w}$ make zero points in V_2^p and t points in V_1^p. Thus, all candidates from D are beaten by c. Regarding the candidates from Z_j, every candidate appears either in $Z_{w,j}$ or in $\overline{Z_{w,j}}$ and thus makes exactly one point and is beaten by c. Clearly, all candidates from $A \cup B$ are also beaten by c. Hence, c is a possible winner for X.

Finally, we briefly discuss that fixing the order within the given subsets of candidates in Figure 9.4 can be done without violating the restriction provided by the partial orders. For v_j in V_2^p such an extension is

$$B > D_w > w_j > \overline{D_w} > \bigcup_{i \neq j} C_i \setminus \{c_i\} > A > Z > \{c\}$$

where, the candidates from $B, D_w, \overline{D_w}, A$, and Z can be fixed in an arbitrary order since there are not any internal constraints in v_j. The remaining candidates from $\bigcup_{i \neq w} C_i \setminus \{c_i\}$ can be ordered such that $C_i \setminus \{c_i\} > C_s \setminus \{c_s\}$ for $i > s$, $i \neq w$, and $s \neq w$ and within $C_i \setminus \{c_i\}$, for every $i \neq w$, the candidates can be ordered according to any extension of v_i^j. A "complete" extension for the votes from V_1^p can be obtained similarly.

"$\Rightarrow$": Consider an extension of X in which c wins. First, by proving the following claim, we show that within V_1^p one instance x_w must be "selected".

Claim: There must be a $w \in \{1, \ldots, q\}$ such that every candidate from $\overline{D_w}$ is assigned to a one-position in every extended vote from V_1^p whereas every candidate from D_w makes zero points in V_1^p.

Proof of Claim: Since there are $3s+4$ one-positions per vote, in V_1^p there are altogether $3st+4t$ one-positions that must be filled. The candidates from Z can take at most $2st+2t$ of them since $|Z| = 2t(s+1)$ and each candidate from Z can make at most one point without beating c. By using some argumentation including the votes from V_2^p, we can show that the candidates from D can take at most $st+t$ of the one-positions in V_1^p in a winning extension: In every vote from V_2^p, by construction, the first $2s+3-k$ positions are assumed by candidates from B and the remaining $s+k+1$ one-positions can only be assigned to candidates from $\bigcup_{1 \le i \le q} C_i \setminus \{c_i\} \cup D$. Since every candidate from $\bigcup_{1 \le i \le q} C_i \setminus \{c_i\}$ shifts $s+1$ candidates from D to the left by assuming a one-position, it directly follows that the total number of one-positions assumed by candidates from D within V_2^p is at least $t(s+1)$. Since $|D| = 2s+2$ and every candidate from D can make at most t points, the candidates from D can take at most $t(2s+2) - t(s+1) = st+t$ of the one-positions in V_1^p in a winning extension.

Summarizing, in a winning extension, in V_1^p at most $3st+3t$ one-positions can be assigned to candidates from $D \cup Z$. Hence, at least t one-positions must be assigned to candidates from A. Furthermore, a candidate a_i from A shifts $3s+3$ candidates from $D \cup Z$ to one-positions if a_i takes a one-position. Thus, at most one candidate from A can take a one-position in an extended vote. It follows that in every vote $v_j \in V_2^p$ exactly one candidate a_i from A must take a one position thereby shifting the candidates from $Z_{i,j} \cup \overline{D_i} \cup \overline{Z_{i,j-1}}$ (or $Z_{i,1} \cup \overline{D_i} \cup \overline{Z_{i,t}}$ for $j=1$) to one-positions.

Now, we show for $1 \le j \le t-1$ that if the candidate $a_w \in A$ takes a one-position in v_j, then a_w also takes a one-position in v_{j+1}. Assume that in v_j, a_w and thus also the candidates from $Z_{w,j}$ take a one-position. As discussed above, in v_{j+1} a candidate from A must shift $s+1$ further candidates from Z_j. Since every candidate from Z can make at most one point, the set of these candidates must be disjoint from $Z_{w,j}$. The only set of candidates fulfilling this is $\overline{Z_{w,j}}$ and is shifted only by a_w. Analogously, if a_w takes a one-position in v_t, then it also must take a one-position in v_1 because of the candidates from Z_t. This finishes the proof of the Claim.

Now, as direct consequence of the Claim, within V_2^p each candidate from D_w can still make t points whereas the candidates from $\overline{D_w}$ cannot make any points without beating c. Hence, in every vote from V_2^p, we can only set candidates from C_w to the one-positions since setting any other candidates would shift a candidate from $\overline{D_w}$. This means that one can extend V_2^p such that, in every vote, k one-positions are assigned to candidates from $C_w \setminus \{c_w\}$ without beating c. Since the partial relations between the candidates in $C_w \setminus \{c_w\}$ are the same in the ith vote of x_w and X and c makes zero points in both cases, a winning extension for X directly gives a winning extension for x_w. □

By using Lemma 9.5 it is easy to verify that the given composition algorithm fulfills all requirements of Definition 9.1. Hence, Theorem 9.7 follows from our overall strategy.

Table 9.1: Overview of results for Possible Winner under k-approval with t partial votes. The left column summarizes the results for the combined parameter t and "number of zero-positions" k'. The right column shows the results with respect to the combined parameter t and k.

$(\mathbf{t}, \mathbf{k}')$	$(\mathbf{t}, \mathbf{k})$
polynomial kernel (Theorem 9.1)	superexponential kernel (Theorem 9.4) no polynomial kernel (Theorem 9.7) 2-approval: polynomial kernel with $O(t^2)$ candidates (Theorem 9.5)
dyn. prog. $O(2^{tk'} \cdot (tk')^{k'} \cdot nm^2)$ (Theorem 9.3)	**?**
search tree $O(2^{t^2k'} \cdot nm^2)$ (Proposition 9.1)	**?**

Theorem 9.7. *For k-approval,* Possible Winner *parameterized by the combined parameter k and "number of partial votes" does not admit a polynomial problem kernel unless* NP $\subseteq$ coNP / poly.

9.3 Conclusion

Our main results are summarized in Table 9.1. Several concrete question arise directly from the results in this chapter.

- Is there a polynomial problem kernel with a linear number of candidates with respect to the "number of partial votes" for Possible Winner for 2-approval?
- For Possible Winner under k-approval, we provided a kernel of superexponential size and showed that presumably no polynomial kernel exists. Clearly, this leaves room for improving the kernel size to some "reasonable" exponential or even subexponential function.
- Are there efficient fixed-parameter algorithms with respect to t when k is constant for Possible Winner under k-approval. In particular, can there be obtained "improved" problem kernels in a similar style as the kernel with a quadratic number of candidates for 2-approval for any constant $k > 2$?
- Can our results be extended to other scoring rules? For example, consider the scoring rule in which there are k candidates receiving two points and k' candidates getting zero points while every remaining candidate gets one point. Intuitively, such rules allow not only to specify some favorites but also to "reject" a set of most disliked candidates. Recall that Possible Winner for this rule is NP-hard for $k = k' = 1$ [14] as well as for all other values of k and k' of at least one [19] (see Chapter 7). This leads to the question whether this problem is fixed-parameter tractable with respect to $(t, k + k')$.

Chapter 10

Conclusion Part II

The second part of the thesis provided a multivariate complexity study of Possible Winner under scoring rules from various perspectives. Summarizing, Chapter 7 contains a dichotomy result (also based on [14, 194]) which explains the influence of the scoring rule to the computational complexity of Possible Winner. Herein, the most technical part was the development of many-one reductions for scoring rules "charting the border to polynomial-time solvability". For example, we showed NP-hardness for 2-approval whereas plurality (1-approval) can be solved in polynomial time. Chapter 8 gives a parameterized complexity analysis with respect to several single parameterizations. The main technical contribution is the proof of the NP-hardness for Possible Winner for Borda in case of three partial votes. We also promoted a new line of research by discussing several parameterizations measuring the amount of incompleteness that might be relevant for future research (see Section 8.3 for concrete questions). Finally, in Chapter 9, for Possible Winner under k-approval voting, we used kernelization to show fixed-parameter tractability for combined parameterizations. This provides one of the few examples for kernelization for voting problems.

We conclude with several remarks, which put our contributions into the context of known results or discuss directions of future research.

Sets of winners. Our results have been stated for the unique-winner case and directly transfer to the cowinner case where, for scoring rules, one allows that several candidates get the maximum score and all of them win. A natural model for multiwinner elections under scoring rules can be described as follows. One has given a positive integer s specifying the number of winners, for example, the size of a board, and the s candidates with most points win. If such a set cannot be determined because there are several candidates with the same score, then one can apply an appropriate tie-breaking rule depending on the situation. Formally, this leads to the following problem.

> Possible s-Multiwinner
> *Given:* A voting correspondence returning s winners, a set C of candidates, a set of partial votes V on C, and a distinguished candidate c.
> *Question:* Is there an extension of V such that c is part of the set of winners?

Herein, a *voting correspondence* denotes a function from a multiset of votes to a subset of s candidates. We briefly discuss how our results apply to the extended setting.

First, all the negative results from this part of the thesis can be adapted in a straightforward way. To this end, in the corresponding hardness reductions one can add $s-1$ candidates that are fixed at "irrelevant" positions in the partial votes and that always must be part of the winner set. In principle, the composition algorithm (Subsection 9.2.2) can also be applied to this special case since it is NP-complete. However, herein one first needs to investigate whether the corresponding problem is fixed-parameter tractable since for the kernelization results from this chapter it is not immediately clear that they carry over to the multiwinner setting.

Second, regarding the algorithmic results, the general search tree algorithm provided in Section 8.3 to show fixed-parameter tractability with respect to "the total number of undetermined pairs" directly works for the extended scenario. This is easy to verify since basically the algorithm enumerates all possible extensions. For the parameter "number of candidates" the fixed-parameter tractability result based on an ILP-formulation (see Section 8.1) also can be transferred to the considered multiwinner setting. As a first step one can branch into all size-s candidate subsets containing the distinguished candidate and then formulate an ILP to check if the considered subset is a winner set.

For k-approval, it seems natural to search for a winning set of size k, that is, to consider the Possible k-Multiwinner problem, leading to the following question.

> Does the Possible k-Multiwinner problem under k-approval admit (polynomial) problem kernels with respect to the combined parameters (t, k') and (t, k)?

Moreover, the investigation of Possible Winner for other multiwinner settings and voting systems is clearly of interest.

Special cases and generalizations. Possible Winner is a fundamental combinatorial problem. Hence, there are many relevant variations, generalizations, and special cases, some of them already studied in the literature.

Let us start with a discussion about special cases "making the problem easier". Besides a huge amount of work on the famous Manipulation problem (see also Subsection 2.2.2), in a very recent work Chevaleyre et al. [54] investigated the setting of adding s candidates. Formally, this is the variant of Possible Winner where one has complete information about the relative orders of all but s candidates and no information about these s candidates at all. In contrast to Possible Winner in general, this variant becomes polynomial-time solvable for Borda. However, for 4-approval and adding three candidates the problem is NP-complete [54]. The computational complexity of some of the remaining cases, like 2-approval and adding more than one candidate, is still open.

One way to get potentially tractable special cases of Possible Winner is to restrict the structure of the input votes. We briefly discuss three such possibilities.

- Xia and Conitzer [194] suggested to investigate the Possible Winner problem for partial votes that must be submitted in a restricted form, more specifically,

corresponding to CP-nets. In this case the negative results do not transfer immediately [194]. Hence it seems a challenging task to identify scenarios for which CP-nets help to design efficient algorithms.

- Another reasonable restriction is to investigate "single-peaked preferences" [33] as suggested by Walsh [193] and further investigated by Faliszewski et al. [100] and very recently by Brandt et al. [41]. Roughly speaking, in case of single-peaked preferences, there is a linear order of the candidates and every voter has a designated "peak" in this order such that the voter's preference for a candidate decreases with the distance from its peak. For weighted votes, Walsh [193] provided evidence that for Possible Winner for some voting systems (e.g., STV) single-peaked preferences still lead to computational hardness. In contrast, Faliszewski et al. [100] showed that for some voting systems (e.g., k-approval) for Manipulation the NP-hardness evaporates. For the case of Possible Winner and unweighted votes (as studied in this work), we are not aware of any results in these directions.

- It seems reasonable to consider restrictions that make sense for specific voting systems. For example, to determine a winner for k-approval under complete information it is sufficient for a voter to partition the candidates into a "winner set" and a "looser set". Under incomplete information, this could be extended by additionally allowing for "a set of undecided" candidates. This model seems likely to drastically decrease the computational complexity (which still needs to be investigated) and still might reflect some natural situations.[1] Note that this problem is also very similar to a setting from Faliszewski [93] in "nonuniform bribery": This bribery variant allows to move points at a given price between voters. Then, given some "moves of points" the prize zero, leads to the scenario described above. Further identifications of such restricted input structures seem to be of general interest.

We end with generalizations of Possible Winner. Regarding concrete problems, Swap Bribery [80] can be considered to "subsume" Possible Winner. Swap Bribery is a variant of Bribery (see also [93, 99]) in which an external agent can pay a voter to change his vote by swapping candidates. Herein, each swap is associated with a specific price. According to [80, Theorem 2], Possible Winner many-one reduces to Swap Bribery for every voting rule. Recently, Dorn and Schlotter [74] provided some fixed-parameter tractability results that show that some of the algorithmic results from this part can be extended to Swap Bribery. In particular, they present an ILP-formulation showing fixed-parameter tractability with respect to the "number of candidates" and kernelization results for combined parameters.

Considering generalizations of votes itself, a well-studied scenario is to consider weighted votes. Here, for all scoring rules except plurality already the Manipulation problem is NP-hard [129]. Hence, this case clearly deserves a multivariate complexity analysis aiming at efficient fixed-parameter algorithms.

[1] In general, this leads to a much less powerful model as the general possible-winner model studied here. For example, for the case of adding s candidates [54] (see also above), a voter might not be able to put any candidate into the "winning set" if he has no information about the candidates to be added although he already has decided about a linear order on all known candidates. In contrast, the general definition of Possible Winner allows such a voter to state this information.

Voting systems with NP-hard winner determination. For voting systems in which the determination of a winner is NP-hard under complete information, Possible Winner is obviously NP-hard as well. Parameterized algorithmics might offer a way to tackle Possible Winner for such voting systems. For example, the first part of this work comprises fixed-parameter algorithms for the winner determination in Kemeny, Dodgson, and Young elections. It is open whether these results carry over to the Possible Winner problem.

Using social choice properties. An interesting line of research is to settle the computational complexity of "classes" of voting rules classified according to social choice properties. For example, Pini et al. [178] showed that for all voting systems fulfilling "monotonicity" and "Independence of Irrelevant Alternatives (IIA)", Possible Winner can be solved in polynomial time (even for weighted votes and either constant or unbounded number of candidates). Similar results for other social choice properties would be clearly desirable.

Counting winning extensions. Recent work [7] proposes a quantitative approach which instead of just investigating the existence of a winning extension counts the number of extensions leading to a designated candidate's victory. To this end, it introduces the counting version of Possible Winner denoted as #-Possible Winner. This approach allows to compare two candidates that are "possible winners". We briefly summarize the results and state some open questions. On the negative side, one encounters #P-hardness results for #-Possible Winner even for plurality and veto. On the positive side, there is a simple sampling algorithm with provable performance guarantee. More specifically, there is a randomized polynomial-time approximation algorithm for the problem of computing the proportion of extensions of a profile where the distinguished candidate wins. For #-Manipulation for k-approval, the work provides a dynamic programming algorithm running in polynomial time if k is a constant. Herein, the fixed-parameter tractability with respect to k is still open. It is also open whether there is a (natural) voting rule for which Manipulation is solvable in polynomial time whereas #-Manipulation becomes #P-hard. Another interesting question regards fixed-parameter tractability with respect to the "number of candidates". This directly leads to the following question of general interest. Does Lenstra's results (see Subsection 1.3.3) hold or can it be modified to work for counting optimal solutions. Note that although "counting problems" are important in general, there are only few works on parameterized algorithmics in this direction. More precisely, two works [165, 112] focus on extending the framework of parameterized complexity to counting problems and several works are concerned with finding enumeration algorithms running in "fpt-time", for example [67, 109, 174].

Final remarks. Whereas our main focus is on worst-case analysis and exact algorithms, there are several interesting and challenging lines of research regarding average-case analysis as well as randomized and approximation algorithms. Clearly, a combination of such approaches with a multivariate complexity analysis is of great interest. Some "easy-to-state" tasks in these directions regard the development of approximation algorithms for restricted scenarios. For example, can one obtain a better approximation guarantee when having a small amount of incompleteness measured by

one of the parameterizations from Section 8.3.

Finally, multivariate algorithmics does not restrict "what" can be chosen as a parameter and it seems impossible to provide a complete "list of parameterizations" for a problem. Hence, the identification of (new) meaningful parameterizations is always of interest. However, in general it is not clear how to measure the "significance" of a specific parameter. This leads the way to data-driven algorithmics. Here, one analyzes typical properties of real-world data in hope to identify data-specific parameterizations with small parameter values.

Part III

Candidate Control

There are different ways for an external agent to influence the outcome of an election. We concentrate on "control" by adding or deleting candidates. We investigate the parameterized complexity of various control problems for different voting systems. To this end, we introduce natural digraph problems that may be of independent interest. They help in determining the parameterized complexity of control for different voting systems including Llull, Copeland, and plurality voting. Devising several parameterized reductions, we settle the parameterized complexity of the digraph and control problems with respect to natural parameters such as adding/deleting a bounded number of candidates or having few voters.

Chapter 11

Candidate control for Copeland and plurality

To *control* an election, an external agent, misleadingly called the *chair* in the literature, can change the voting procedure to reach certain goals. For example, a typical question is whether the chair can make his/her favorite candidate a winner by deleting a certain number of candidates. Traditionally, considered types of control are adding, deleting, or partitioning candidates or voters [13]. Furthermore, one distinguishes between *constructive control* (CC), where the chair aims at making a distinguished candidate a winner, and *destructive control* (DC), that is, the chair wants to prevent a distinguished candidate from winning [131].

The investigation of the computational complexity of control problems goes back to Bartholdi, Tovey and Trick [13]. They defined a voting system to be *immune* against a type of control if it is never possible for the chair to change a non-winner candidate to become a winner candidate, otherwise it is *susceptible* for the considered kind of control. Unfortunately, commonly used voting systems are susceptible to some types of control. For example, plurality voting is even susceptible to all standard types of control. Thus, Bartholdi et al. [13] suggested computational hardness as a favorable property of voting systems if immunity is not guaranteed. Here, one classically distinguishes between *resistant*, that means, controlling the election is NP-hard, and *vulnerable*, that is, controlling the election can be accomplished in polynomial time. Note that the term "resistant" may be misleading in the sense that it does only imply hardness for a worst-case scenario. Nevertheless, it seems interesting to investigate whether there are efficient strategies for control in general.

A series of publications [13, 99, 131] provides a complete picture of the classical computational complexity for eleven basic types of control for the standard voting systems approval, plurality, Condorcet, and Copeland$^{\alpha}$ for all rational values of α in the range of $[0, 1]$. Additionally, Hemaspaandra et al. [128] showed that hybrid elections can lead to stronger resistance results for electoral control. Further work [88, 89, 91, 90] considers control for two specific hybrid systems combining approval voting and systems based on linear preferences. Liu et al. [158] considered the parameterized complexity of some types of control for maximin. Recently, Faliszewski et al. [95]

introduced the extended scenario of "multimode control attacks", that is, the chair is allowed to use various kinds of attacks like deleting candidates and adding votes simultaneously.

We focus on *candidate control*, that is, either deleting or adding candidates for plurality and Copeland$^\alpha$ voting described in the following.[1]

Copeland$^\alpha$ voting for rational values of α in the range of $[0, 1]$ is based on pairwise comparisons between candidates: A candidate wins the *pairwise head-to-head contest* against an other candidate if it is better positioned in more than half of the votes. The winner of a head-to-head contest is awarded one point and the loser receives no point. If two candidates are tied, both candidates get α points. A *Copeland$^\alpha$ winner* is a candidate with the highest score. Faliszewski et al. [97] devoted their paper to the two important special cases $\alpha = 0$, denoted as *Copeland*, and $\alpha = 1$, denoted as *Llull*. Copeland$^\alpha$ elections are used in various settings. For instance, in sport tournaments, like chess or in football leagues, the teams or players can be considered as candidates. The value of α depends on the type of sport, for example, $\alpha = 1/3$ for the German soccer league. Throughout this chapter, α always denotes a rational number within $[0, 1]$.

We briefly summarize known results about the computational complexity of candidate control for plurality and Copeland$^\alpha$. Copeland$^\alpha$ voting is resistant to constructive candidate control and vulnerable for destructive candidate control [99]. Plurality voting is resistant to constructive and destructive control by adding and by deleting candidates [13, 131]. Regarding parameterized complexity, Faliszewski et al. [99] considered control of Copeland$^\alpha$ voting with respect to the parameters "number of candidates" and "number of votes" for constructive and destructive control in the eleven standard control scenarios. For control by adding and deleting candidates they obtained fixed-parameter tractability with respect to the parameter "number of candidates" for all considered scenarios. The parameterized complexity with respect to the parameter "number of votes" was left open.

In this chapter, we investigate the parameterized complexity with respect to the natural problem-specific parameterizations "number of added/deleted candidates". Since it seems plausible that the chair can add or delete only few candidates without raising suspicion, the existence of efficient fixed-parameter algorithms for such parameters would yield a general control strategy for natural voting scenarios. Note that the goal of many publications is to show that, if control is not impossible, it is at least computationally hard (often showing NP-hardness). However, as noted by Conitzer et al. [65], such hardness results lose relevance if there are efficient fixed-parameter algorithms for realistic settings. We devise W[1]-hardness or W[2]-hardness results for most considered settings. A crucial step to obtain these results is to investigate closely related digraph problems (which might be of interest on their own). In the following section, we introduce these digraph problems and discuss their relations with the control problems. Then we will provide an overview of our results for all considered problems and the further organization of this chapter (Subsection 11.1.3).

[1] We do not include control by partitioning the set of candidates in the definition of the term "candidate control".

11.1 Candidate control and related digraph problems

Before introducing some new digraph problems, we give a formal definition of the control problems.

11.1.1 Control problems

As in Part II, we only consider the unique-winner case, but all our results can be easily modified to work for the winner case as well. We focus on control by adding candidates (AC) or deleting candidates (DC). For all rational $\alpha \in [0, 1]$, we can define the decision problems of constructively controlling a Copeland$^\alpha$ election by deleting and adding candidates as follows:

> CC-DC-Copeland$^\alpha$
> *Given:* A set C of candidates, a multiset V of votes over C, a distinguished candidate $c \in C$, and an integer $k \geq 1$.
> *Question:* Is there a subset $C' \subseteq C$ of size at most k such that c is the unique Copeland$^\alpha$ winner in the election $(V, C \backslash C')$?

> CC-AC-Copeland$^\alpha$
> *Given:* Two disjoint sets C, D of candidates, a multiset V of votes over $C \cup D$, a distinguished candidate $c \in C$, and an integer $k \geq 1$.
> *Question:* Is there a subset $D' \subseteq D$ of size at most k such that c is the unique Copeland$^\alpha$ winner in the election $(V, C \cup D')$?

In general, the first two letters of the name of a problem stand for constructive or destructive control (CC/DC). The following two letters stand for the kind of modification (AC/DC) and are followed by the name of the considered voting system. The control problems for plurality voting and for destructive control are defined analogously (see for example [99, 131]).[2] Note that, in contrast to the version of Possible Winner by adding s candidates [54] (see also Section 10), here we have complete information about the candidates that can be added.

Next, we introduce some digraph problems which are closely related to candidate control in Copeland and Llull elections.

11.1.2 Digraph problems

A Copeland or Llull election can be depicted by a digraph where the candidates are represented as vertices and there is an arc from vertex c to vertex d if and only if the corresponding candidate c defeats the corresponding candidate d in the head-to-head contest. Obviously, the Copeland score of a candidate equals the outdegree of the corresponding vertex and, thus, a Copeland winner corresponds to a vertex with maximum outdegree. The Llull score of a candidate c in an election with m candidates is $m - 1$ minus "the number of candidates that beat c in the pairwise head-to-head contest". Thus, a Llull winner corresponds to a vertex with minimum indegree. Naturally, the deletion/addition of a vertex one-to-one corresponds to the

[2] There is an other version of control by adding candidates [13, 97] in which one asks whether it is possible to control an election by the addition of an unlimited number of candidates. We do not consider this version here.

deletion/addition of a candidate in the election. These observations motivate the introduction of the following digraph problems.

> Max-Outdegree Deletion (MOD)
> *Given:* A digraph $D = (W, A)$, a distinguished vertex $w_c \in W$, and an integer $k \geq 1$.
> *Question:* Is there a subset $W' \subseteq W \setminus \{w_c\}$ of size at most k such that w_c is the only vertex that has maximum outdegree in $D[W \setminus W']$?

Analogously, given a directed graph, a distinguished vertex, and a positive integer k, Min-Indegree Deletion (MID) asks for a set of at most k vertices whose removal makes the distinguished vertex to be the only vertex with minimum indegree. We say that MID correspond to constructive control by deleting candidates for Llull voting and MOD correspond to constructive control by deleting candidates for Copeland voting. The problems for adding vertices are defined as follows:

> Min-Indegree Addition (MIA)
> *Given:* A digraph $D = (W, A)$ with vertex set $W = \mathcal{C} \uplus \mathcal{N}$, a distinguished vertex $c \in \mathcal{C}$, and an integer $k \geq 1$.
> *Question:* Is there a subset $\mathcal{N}' \subseteq \mathcal{N}$ of at most k vertices such that c is the only vertex of minimum indegree in $D[\mathcal{C} \cup \mathcal{N}']$?

> Max-Outdegree Addition (MOA)
> *Given:* A digraph $D = (W, A)$ with vertex set $W = \mathcal{C} \uplus \mathcal{N}$, a distinguished vertex $c \in \mathcal{C}$, and an integer $k \geq 1$.
> *Question:* Is there a subset $\mathcal{N}' \subseteq \mathcal{N}$ of at most k vertices such that c is the only vertex of maximum outdegree in $D[\mathcal{C} \cup \mathcal{N}']$?

For the addition problems we have that MIA corresponds to constructive control by adding candidates for Llull voting and MOA corresponds to constructive control by adding candidates for Copeland voting.

Since the deletion/addition of a candidate one-to-one corresponds to the deletion/addition of a vertex, every instance of a control problem can be transformed to an equivalent instance of the corresponding digraph problem. More specifically, a distinguished candidate can become the only winner of a Copeland election by deleting/adding k candidates if and only if the corresponding vertex can become the only vertex with maximum outdegree by deleting/adding k vertices in the corresponding digraph. In the same way, a distinguished candidate can become the only winner of a Llull election by deleting/adding k candidates if and only if the corresponding vertex can become the only vertex with minimum indegree by deleting/adding k vertices. This directly provides parameterized reductions from the control problems to the corresponding digraph problems with respect to the parameters number of deleted/added candidates and vertices, respectively. In the following, we show how the opposite reductions can be obtained.

A digraph $D = (W, A)$ is *encoded* in an election (V, C) if the outcomes of the pairwise head-to-head contests reflect the arcs of the digraphs. That is, the candidate set is given by $C := \{c_i \mid w_i \in W\}$ and candidate c_i defeats candidate c_j if and only if $(w_i, w_j) \in D$. An election encoding a given digraph can be constructed in polynomial time [166]: For every arc $(w_i, w_j) \in D$, add the two votes $c_i > c_j > C'$

Table 11.1: Parameterized complexity of Max-Outdegree Deletion (MOD) and Min-Indegree Deletion (MID). W[2]-membership is given in Theorem 11.6, the other results are from [1]Theorem 11.1, [2]Proposition 11.2, [3]Theorem 11.2, [4]Proposition 11.3. Clearly, it does not make sense to consider tournaments with degree constraints.

parameters	# deleted vertices k		maximum degree d		(k, d)	
problems	MOD	MID	MOD	MID	MOD	MID
general digraphs	W[2]-c[1,3]	W[2]-c[3]	NP-c, $d \geq 3$[1]	FPT[2]	FPT[4]	FPT[2]
acyclic digraphs	W[2]-c[1]	P[2]	NP-c, $d \geq 3$[1]	P[2]	FPT[4]	P[2]
tournaments	W[2]-c[3]	W[2]-c[3]	-	-	-	-

Table 11.2: Results in boldface are new. The results for Copeland$^\alpha$ hold for all $0 \leq \alpha \leq 1$. The W[2]-hardness results for CC-AC-Plurality and DC-AC-Plurality follow from the NP-completeness proofs [13, 131]. The polynomial-time (P) results are from [99].

	Copeland$^\alpha$		plurality	
	CC	DC	CC	DC
Adding Candidates (AC)	**W[2]-c**	P	W[2]-h	W[2]-h
Deleting Candidates (DC)	**W[2]-c**	P	**W[2]-h**	**W[1]-h**

and $\overline{C'} > c_i > c_j$ with $C' := C \setminus \{c_i, c_j\}$ to V. In these two votes, c_i beats c_j and all other pairs of candidates are tied. By this, we have a voting system with $2 \cdot |A|$ votes encoding D. The following proposition follows directly.

Proposition 11.1. Max-Outdegree Deletion *(*Min-Indegree Deletion*) and* CC-DC-Copeland *(*CC-DC-Llull*) are FPT-equivalent with respect to the parameters "number of deleted vertices" and "number of deleted candidates", respectively.* Max-Outdegree Addition *(*Min-Indegree Addition*) and* CC-AC-Copeland *(*CC-AC-Llull*) are FPT-equivalent with respect to the parameters "number of added vertices" and "number of added candidates", respectively.*

Finally, note that in tournaments MOD and MID coincide. Considering this from the viewpoint of the corresponding voting problems, this is fairly obvious: Copeland$^\alpha$ elections differ only in the way in which ties are evaluated, and, in an election corresponding to a tournament, there is no tie between any pair of candidates.

11.1.3 Contributions and organization

We provide a first study of the newly introduced digraph problems that are closely related to the considered control problems. In Section 11.2, we investigate the computational complexity of MOD and MID (as well as MIA and MOA) for several special graph classes and parameters, providing a differentiated picture of their parameterized complexity including algorithms and intractability results (see Table 11.1). The main technical achievement of this section is to show that MOD and MID are W[2]-complete in tournaments. One interesting observation is that, although MOD and MID seem to

be very similar, their (parameterized) complexity varies for different graph classes for several parameterizations (Table 11.1). Some of the considered special cases and parameterizations of the digraph problems map to realistic voting scenarios with presumably small parameters. Based on these connections and by giving new parameterized reductions, in Section 11.3, we provide an overview of parameterized hardness results for control problems with respect to the "number of deleted/added candidates" (Table 11.2). Surprisingly, for plurality voting, which can be considered as the "easiest" voting system in terms of winner evaluation and for which the Manipulation problem can be solved optimally by a simple greedy strategy [65], all kinds of candidate control are intractable from this parameterized point of view. The reductions used for the digraph problems often rely on similar ideas. In contrast, the parameterized reductions used for plurality voting require new approaches. Regarding the structural parameter "number of votes", we answer an open question of Faliszewski et al. [98] for Llull and Copeland voting by showing that even for a constant number of votes candidate control remains NP-hard (see Subsection 11.3.2). For this, we use a simple but elegant method based on the considered digraph problems.

11.2 Parameterized complexity of the digraph problems

This section is concerned with the parameterized complexity of the four introduced digraph problems with respect to the parameterizations "number of deleted vertices" k and "maximum degree" d, for different classes of graphs. Our results for deleting vertices are summarized in Table 11.1 and the results for adding candidates are given in Section 11.2.2 (see Table 11.3). The next two subsections give W[2]-hardness and algorithmic results for the vertex deletion and the vertex addition problems. The W[2]-membership for all considered problems is discussed at the end of this section. Note that some of the constructions given in this section are reused in Section 11.3.2.

The following two problems are used for reductions in this section.

> Hitting Set (HS)
> *Given:* A subset family $\mathcal{F} = \{F_1, F_2, \ldots, F_m\} \subseteq 2^S$ of a base set $S = \{s_1, s_2, \ldots, s_n\}$ and an integer $k \geq 1$.
> *Question:* Is there a subset $S' \subseteq S$ of size at most k such that for every $1 \leq i \leq m$ we have $S' \cap F_i \neq \emptyset$?

The set S' is called a *hitting set*. The Hitting Set problem is known to be W[2]-complete [76]. Note that Hitting Set is NP-complete even if every subset has size two and every element occurs in exactly three subsets [118]. This restriction of Hitting Set is denoted as 3X-2-Hitting Set.

> Independent Set (IS)
> *Given:* An undirected graph $G = (U, E)$ and an integer $k \geq 1$.
> *Question:* Is there a subset $U' \subseteq U$ of size at least k such that no two vertices in U' are adjacent?

The set U' is called an *independent set*. The Independent Set problem is W[1]-complete on general graphs [76] and NP-complete even when restricted to graphs with maximum degree 3 [118]. We call this special case $3d$-Independent Set($3d$-IS).

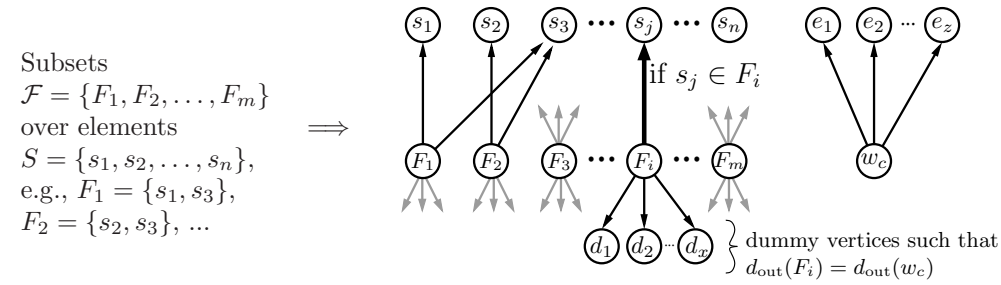


Figure 11.1: Parameterized reduction from a Hitting Set-instance (left) to an MOD-instance (right).

11.2.1 Vertex deletion

For Max-Outdegree Deletion we show the following.

Theorem 11.1. Max-Outdegree Deletion *is W[2]-hard with respect to the parameter "number of deleted vertices" in acyclic digraphs and NP-complete in acyclic digraphs with maximum degree three.*

Proof. Given a Hitting Set instance $H = (\mathcal{F}, S, k)$ with base set S and subset family $\mathcal{F}$, we construct the following digraph $D = (W, A)$ (see Fig. 11.1). The vertex set is given by $W := \{w_c\} \cup W_S \cup W_{\mathcal{F}} \cup D_w \cup \bigcup_{i=1}^{m} D_i$. Herein, w_c is the vertex we would like to become maximum degree vertex. Furthermore, we have a *subset-vertex* for every subset and an *element-vertex* for every element, that is, $W_{\mathcal{F}} := \{F_i' \mid F_i \in \mathcal{F}\}$ and $W_S := \{s_i' \mid s_i \in S\}$. The remaining *dummy-vertices* are specified as follows: Let z denote the maximum size over all subsets, that is, $z := max_{i \in \{1,2,\ldots,m\}} |F_i|$, then D_w consists of z vertices (needed as out-neighbors for w_c) and for every F_i' we have a (possibly empty) set D_i which contains $z - |F_i|$ further dummy-vertices. The arc set is given by $A := \{w_c\} \times D_w \cup \bigcup_{i=1}^{m} (\{F_i'\} \times D_i) \cup A_{\mathcal{F},S}$, where $A_{\mathcal{F},S}$ contains arcs from the subset-vertices to the element-vertices as follows: $A_{\mathcal{F},S} := \bigcup_{i=1}^{m} \{F_i'\} \times \{s_j' \mid s_j \in F_i\}$.

> *Claim*: H has a hitting set of size k if and only if vertex w_c can become the only maximum outdegree vertex by deleting k vertices.

"$\Rightarrow$": Observe that w_c and all subset-vertices of $W_{\mathcal{F}}$ have outdegree b and all other vertices have outdegree zero. Hence, given a hitting set $S' \subseteq S$, after the deletion of the corresponding element-vertices, in the resulting graph all subset-vertices with exception of w_c have outdegree at most $b - 1$.

"$\Leftarrow$": Given a solution $W' \subseteq W \backslash \{w_c\}$ for the MOD-instance. If W' contains only element-vertices of W_S, then the corresponding elements constitute a hitting set: In order to make w_c the vertex with maximum outdegree we have to ensure that the outdegree of every subset-vertex of $W_{\mathcal{F}}$ is decreased by one, that is, for every subset-vertex at least one neighboring element-vertex must be included in the solution. Hence, it remains to show that we can transform any solution into a solution which consists solely of element-vertices. To this end, assume that the solution contains a dummy-vertex $d \in D_i$. Deleting d from the graph decreases only the outdegree of F_i'. Hence, we can instead delete from the graph an element-vertex s_j' with $s_j' \in F_i$, which also

decreases the outdegree of vertex F_i' and has no effect on the outdegree of w_c. With a similar argument we can assume that a minimum solution does not contain any subset-vertex of $W_{\mathcal{F}}$.

The resulting digraph is acyclic (see Fig. 11.1), which gives the first part of the theorem. The second part follows by applying the described reduction from 3X-2-Hitting Set instead of Hitting Set. Then, since every subset contains exactly three elements we do not need any dummy-vertices and the outdegrees of the corresponding subset-vertices and the distinguished vertex are bounded by 3. Furthermore, the indegree of every element-vertex is two since every element only occurs in two subsets. Altogether, the NP-hardness for bounded degree follows. □

In contrast to the results for MOD, for MID we can state the following.

Proposition 11.2. Min-Indegree Deletion *can be solved in linear time in acyclic digraphs. In general digraphs, it is fixed-parameter tractable with respect to the parameter "indegree of the distinguished vertex w_c".*

Proof. First, in a non-empty acyclic digraph there exists at least one vertex of indegree zero. Thus, the distinguished vertex w_c must have indegree zero to be the minimum indegree vertex. Hence, one can iteratively delete all other vertices with indegree zero. Using a topological ordering of an acyclic digraph, this can be done in linear time.

Second, the parameterized algorithm for MID with respect to the "indegree of the distinguished vertex" works as follows. If for an MID-instance one knows which in-neighbors of the distinguished vertex w_c are part of a minimum-cardinality solution, then the problem becomes trivial: One can delete these vertices and extend the resulting partial solution to a minimum-cardinality solution as follows. One iteratively adds all vertices of indegree smaller than the (new) indegree of w_c to the solution since all vertices of indegree smaller than the distinguished vertex must be deleted. Hence, exhaustively trying all subsets of in-neighbors of w_c yields an algorithm with running time $O(2^{d_{\text{in}}(w_c)} \cdot |W|^2)$. □

Intuitively, for the "hard" MOD problem the approach given for MID fails due to the following reason. Even in the case that we knew which neighboring vertices of the distinguished vertex w_c are part of the solution, in order to eliminate a vertex w' with higher outdegree we have to decide whether it is better to remove vertex w' or out-neighbors of it. Indeed, according to Theorem 11.1, MOD is NP-hard in digraphs with degree bounded by three. However, the following proposition shows that with the combined parameter maximum vertex degree d and number of deleted vertices k the problem becomes fixed-parameter tractable.

Proposition 11.3. Max-Outdegree Deletion *is fixed-parameter tractable with respect to the combined parameter "outdegree $d_{out}(w_c)$ of the distinguished vertex" and "number of deleted vertices k".*

Proof. We give a simple branching strategy. If w_c is not the only vertex with maximum outdegree, then we can determine in polynomial time a vertex $u \in W \setminus \{w_c\}$ with outdegree at least $d_{\text{out}}(w_c)$. Then, to make w_c the maximum outdegree vertex, one must either delete u or an out-neighbor of u. More specifically, consider an arbitrary set $N \subseteq N_{\text{out}}(u)$ with $|N| = d_{\text{out}}(w_c)$. Clearly, if one does not delete u itself, then one has to delete at least one vertex from N. Since we do not know in advance which

choice leads to a solution, we branch into all possibilities (at most $d_{\text{out}}(w_c) + 1$) to delete a vertex in $(N \cup \{u\}) \setminus \{w_c\}$. In each branch we decrease the parameter k by one (since we have deleted a vertex) and recursively solve the created subinstance. The recursion stops if w_c has become the only vertex with maximum outdegree or $k \leq 0$. For the running time note that at each level of the recursion for every subinstance we branch into at most $d_{\text{out}}(w_c) + 1$ cases and that the recursion stops at the kth level. Moreover, at every level of the recursion for every subinstance all changes are clearly doable in polynomial time. Thus, the total running time is bounded by $(d_{\text{out}}(w_c) + 1)^k \cdot |W|^{O(1)}$ □

As we will discuss in Section 11.3, tournaments naturally occur in the context of voting systems. Hence, in the following, we investigate the parameterized complexity of MOD restricted to tournaments. Recall that in this case MOD and MID coincide. The following theorem is based on a parameterized reduction from the W[2]-complete Dominating Set problem [76].

> Dominating Set (DS)
> *Given:* An undirected graph $G = (U, E)$ and an integer $k \geq 1$.
> *Question:* Is there a size-k subset $S \subseteq U$ such that every vertex $u \in U \setminus S$ has a neighbor in S?

The reduction shows that MOD and MID are W[2]-hard (and NP-hard) in tournaments. Note that other prominent NP-complete problems such as Hamilton Path are polynomial-time solvable when restricted to tournaments [9].

Theorem 11.2. Max-Outdegree Deletion *and* Min-Indegree Deletion *are W[2]-hard with respect to the parameter "number of deleted vertices" if the input graph is a tournament.*

Proof. We develop a parameterized reduction from Dominating Set to MOD in tournaments. The hardness for MID follows by the hardness of MOD since in tournaments both problems coincide.

The basic idea of the reduction is as follows. We construct an MOD-instance in which, aside from sets of further dummy vertices denoted by F and D, there are two copies of the vertices in the DS-instance, denoted by $\mathcal{N}$ and U', and a vertex c which we would like to become the maximum outdegree vertex. The neighborhood structure of the DS-instance is encoded in the arcs between $\mathcal{N}$ and U'. That is, we have an arc from a vertex in U' to a vertex in $\mathcal{N}$ if the respective vertices are neighbors (or the same vertex) in the DS-instance. An illustration of the resulting MOD-instance is shown in Fig. 11.2. Moreover, we set the arcs between all other vertices such that the following conditions are fulfilled. First, the distinguished vertex c has the same outdegree as the vertices in U'. Second, in order to decrease the outdegree of the vertices in U' below the outdegree of c by deleting k vertices, we are enforced to choose vertices from $\mathcal{N}$ such that every vertex in U' looses at least one out-neighbor. Hence, the chosen vertices correspond to a dominating set in the input instance. In the following we give the formal construction.

To simplify the proof, we assume that the graph of the DS-instance has an odd number of vertices and that $k < n$. These assumptions clearly do not limit the generality of the reduction. Given a DS-instance $(G = (U, E), k)$ where $U = \{u_1, u_2, \ldots, u_n\}$

with n odd, we construct a tournament graph $T = (W, A)$ as follows. The set of vertices is $W := \{c\} \uplus U' \uplus \mathcal{N} \uplus D \uplus F$ where c is the vertex that we would like to become maximum outdegree vertex. Furthermore,

- $U' := \{u'_i \mid i = 1, \ldots, n\}$ simulates that every vertex has to be dominated and
- $\mathcal{N} := \{n_i \mid i = 1, \ldots, n\}$ simulates that every vertex can dominate its neighborhood.
- The set $D := \{d_i \mid i = 1, \ldots, n\}$ ensures that only vertices of $\mathcal{N}$ can be deleted.
- Finally, we need a set of dummy vertices $F := \{f_i \mid i = 1, \ldots, 20n + 1\}$ that are used to "set" the outdegrees of the other vertices in an appropriate way.

Next, we describe the construction of the arc set A. The goal of the construction is to ensure that c has the same outdegree as all vertices in U' and to decrease the outdegree of a vertex $u'_i \in U'$ only vertices from $\mathcal{N}$ that correspond to the closed neighborhood of u_i can be deleted. See Fig. 11.2 for an illustration. Within $\mathcal{N}$ we can set the arcs such that every vertex has exactly $\lfloor n/2 \rfloor$ out-neighbors inside $\mathcal{N}$ [98, Lemma 3.4]. Analogously, we set the arcs within U', F, and D. Moreover, we add the following arcs between $c, D, \mathcal{N}$, and U' to the arc set A:

- $\{c\} \times U'$ and $\{c\} \times D$ and $\mathcal{N} \times \{c\}$,
- $D \times (U' \cup \mathcal{N})$, and
- $\{u'_i\} \times \{n_j \mid u_j \in N[u_i]\}$ and $\{n_j \mid u_j \in U \setminus N[u_i]\} \times \{u'_i\}$ for $i = 1, \ldots, n$.

Finally, we describe the construction of the arcs between the dummy vertices in F and the vertices outside of F. To this end, we partition F into three sets, namely, $F_u := \{f_1, f_2, \ldots, f_{2n-1}\}$, $F_c := \{f_{2n}, f_{2n+1}, \ldots, f_{2n+\lfloor n/2 \rfloor - k}\}$, and $F_r := F \setminus (F_u \cup F_c)$. Note that $|F_u| = 2n-1$ and $|F_c| = \lfloor n/2 \rfloor - k+1$. As a consequence, we have that $|F_r| > 17n$. Moreover, for $1 \leq i \leq n$ let $F_u^i := \{f_1, f_2, \ldots, f_{2n-k-|N[u_i]|+1}\}$. We add the following arcs to A.

- $\{u'_i\} \times F_u^i$ and $(F_u \setminus F_u^i) \times \{u'_i\}$ for $i = 1, \ldots, n$,
- $F_u \times (\{c\} \cup D \cup \mathcal{N})$,
- $\{c\} \times F_c$ and $F_c \times (N \cup D \cup U')$, and
- $(\{c\} \cup \mathcal{N} \cup D \cup U') \times F_r$.

This completes the construction of the tournament. By counting the out-neighbors of every vertex one can verify that the following conditions hold (herein, "$a >> b$" means that a is greater than $b + k$):

$$
\begin{array}{lcll}
d_{\text{out}}(c) & = & 2n + \lfloor n/2 \rfloor + |F_R| - k + 1, & \\
d_{\text{out}}(u'_i) & = & 2n + \lfloor n/2 \rfloor + |F_R| - k + 1 & \text{for all } u'_i \in U', \\
d_{\text{out}}(d_i) & = & 2n + \lfloor n/2 \rfloor + |F_R| & \text{for all } d_i \in D, \\
d_{\text{out}}(n_i) & \leq & n + \lfloor n/2 \rfloor + |F_R| & \text{for all } n_i \in \mathcal{N}, \text{ and} \\
d_{\text{out}}(f) & \leq & (|F| - 1)/2 + 3n = 10n + 3n << |F_r| & \text{for all } f \in F.
\end{array}
$$

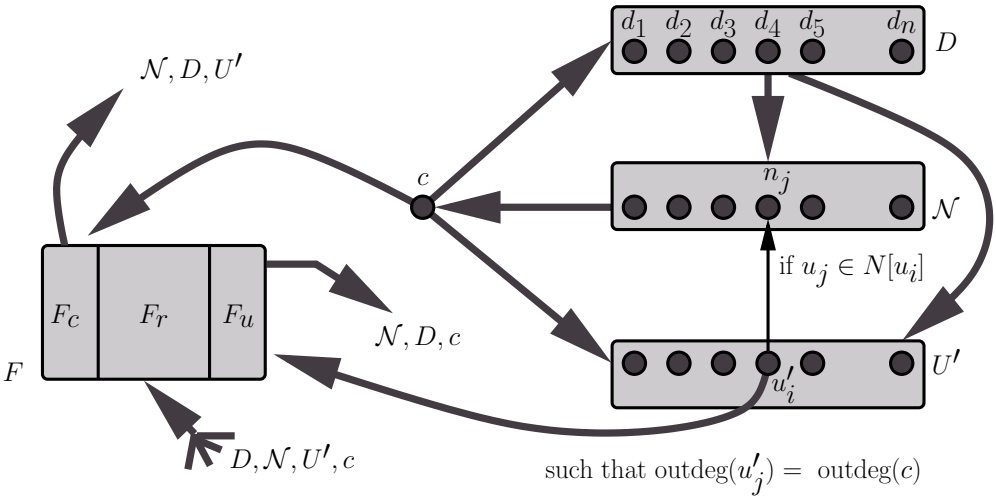


Figure 11.2: Construction of the tournament in the proof of Theorem 11.2. The arcs between the vertices in a shaded box are allocated such that every vertex has outdegree $\lfloor n/2 \rfloor$ or $\lfloor |F|/2 \rfloor$, respectively. The bold arrows indicate bundles of arcs.

In particular, this means that the outdegree of c equals the outdegree of every vertex in U'. Moreover, the outdegree of a vertex in D is by $k-1$ greater than the outdegree of c and the outdegree of every other vertex is by more than k smaller than the outdegree of c. In summary, this means that in order to make c the only vertex of maximum outdegree by the deletion of at most k vertices it remains to ensure that the outdegree of every vertex in U' and D becomes smaller than the outdegree of c. Now, we prove the correctness of the reduction.

> *Claim*: There is a dominating set of size k iff c can become the only vertex with maximum outdegree by deleting k vertices.

"$\Rightarrow$": Let S be a dominating set of size at most k. We show that by deleting all vertices of $W' := \{n_i \in \mathcal{N} \mid u_i \in S\}$, vertex c becomes the only vertex of maximum outdegree. Clearly, the deletion of W' does not affect the outdegree of c. However, the outdegree of every $u'_i \in U'$ is decreased by at least one (by the deletion of a vertex n_j with $u_j \in N[u_i]$) and the outdegree of every $d_i \in D$ is decreased by k. Then we have $d_{\text{out}}(c) - 1 = d_{\text{out}}(d_i) \geq d_{\text{out}}(u_i) > d_{\text{out}}(n_i) > d_{\text{out}}(f_i)$ and, therefore, c is the only vertex of maximum outdegree.

"$\Leftarrow$": First, we show that every solution W' of size k for the MOD-instance contains only vertices from $\mathcal{N}$, that is, $W' \subseteq \mathcal{N}$. This relies on the fact that the difference between the outdegree of c and the outdegree of any $d_i \in D$ is exactly $k-1$. In order to make c the only vertex with maximum outdegree we have to decrease the difference for every $d_i \in D$ by every of the k deletion operations. As we cannot increase the outdegree of c, the deletion of every vertex has to decrease the outdegree of every vertex in D while it must not decrease the outdegree of c. We show that only vertices in $\mathcal{N}$ fulfill these requirements. The deletion of a vertex $x \in D \cup U' \cup F_r \cup F_c$, decreases the outdegree of c. Furthermore, the deletion of a vertex in F_u does not decrease the

Table 11.3: Parameterized complexity of Max-Outdegree Addition (MOA) and Min-Indegree Addition (MIA). All W[2]-membership results are given in Theorem 11.6. The remaining results are given in [1] Theorem 11.3, [2]Theorem 11.4, [3]Proposition 11.4, [4]Proposition 11.5, [5]Theorem 11.5.

parameters	# edited vertices k		maximum degree d		(k, d)	
problems	MIA	MOA	MIA	MOA	MIA	MOA
general digraphs	W[2]-c[1,5]	W[2]-c[2,5]	NP-c, $d \geq 4$[1]	FPT[3]	FPT[4]	FPT[3]
acyclic digraphs	W[2]-c[1]	W[1]-h[2]	NP-c, $d \geq 4$[1]	FPT[3]	FPT[4]	FPT[3]
tournaments	W[2]-c[5]	W[2]-c[5]	-	-	-	-

outdegree of a vertex in D. Hence, the only vertices whose deletion decreases the outdegree of a vertex in D and does not decrease the outdegree of c are the vertices in $\mathcal{N}$. Now, given a solution $W' \subseteq \mathcal{N}$, the deletion of W' does not affect the outdegree of c. Furthermore, for every vertex $u'_j \in U'$ at least one out-neighbor n_i must be deleted in order to ensure that the outdegree of u'_j is less than the outdegree of c. Since an out-neighbor n_i of a vertex u'_j corresponds to a vertex u_i that dominates u_j in G, the set $\{u_i \mid n_i \in W'\}$ is a dominating set in G. □

11.2.2 Vertex addition

In the following, we describe how to obtain similar results for the digraph problems by adding vertices. Table 11.3 provides an overview of our results. Here, the problems seem to be even computationally harder than the deletion problems. For example, the acyclicity of the digraph does not help for solving both of the problems. Also the constructions given within the reductions are less involved (especially for the tournament case). Intuitively, this is due to the fact that one can easily "encode" much information in the subset of vertices that can be added.

Theorem 11.3. Min-Indegree Addition *is W[2]-complete with respect to the parameter "number of added vertices" in acyclic digraphs and NP-complete in acyclic digraphs with maximum degree four.*

Proof. The theorem is based on a reduction from Hitting Set. The construction is illustrated in Fig. 11.3. Herein, the vertices that can be added are marked by a shaded box. These vertices correspond to the elements in S. The distinguished vertex w_c and all the "subset-vertices" F_i have indegree zero before the addition of any "element-vertex." All other vertices have indegree at least one. In particular, the binary tree consists of dummy vertices that ensure that each s_i has indegree at least 1. As discussed below, the binary tree structure is useful for the bounded-degree case. Adding an element-vertex s_j in the digraph has the effect that for all subset-vertices corresponding to the subsets containing s_j the indegree is increased above the indegree of the distinguished vertex w_c, that is, the corresponding subsets are "hit" in the Hitting Set-instance. Hence, a hitting set one-to-one corresponds to a solution of the constructed Min-Indegree Addition-instance. Clearly, the constructed graph is acyclic and the first part of the theorem follows. To see the NP-hardness for $d \geq 4$, we

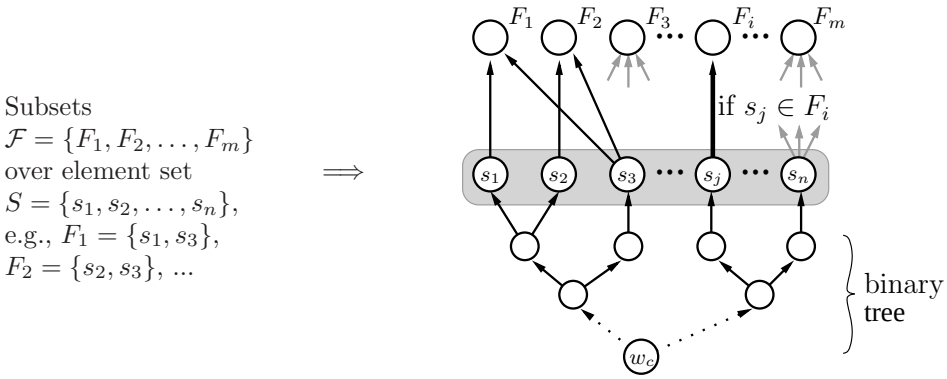


Figure 11.3: Parameterized reduction from a Hitting Set-instance (left) to an Min-Indegree Addition-instance (right).

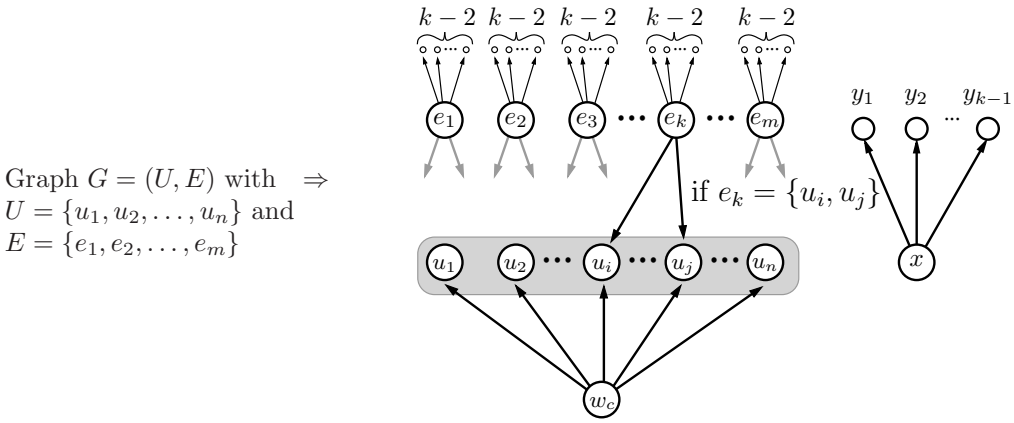


Figure 11.4: Parameterized reduction from Independent Set (left) to Max-Outdegree Addition (right).

reduce from 3X-2-Hitting Set. Then, the constructed graph has maximum degree four. More precisely, every element-vertex has at most three out-going and one in-going arcs, that is, degree four, and all other vertices have degree at most two. This settles the second statement of the theorem. □

Theorem 11.4. Max-Outdegree Addition *is W[1]-hard with respect to the parameter "number of added vertices" in acyclic digraphs.*[3]

Proof. This theorem follows by a reduction from Independent Set. The construction is given in Fig. 11.4. Herein, the vertices that can be added are marked by a shaded box. These vertices correspond to the vertices of the graph of the IS-instance. Each such vertex u_i has in-going arcs from the distinguished vertex w_c and from an "edge-vertex" e_k if u_i is incident to e_k. Moreover, every edge-vertex has $k-2$ further

[3]The corresponding theorem in the journal paper [30] contains an error, stating W[2]-hardness instead of W[1]-hardness.

out-neighbors and there exists a vertex x with outdegree $k-1$. Note that before the addition of any vertex the outdegree of the distinguished vertex w_c is zero. Hence, in order to increase the outdegree of w_c above the outdegree of x one has to add at least k vertices. However, for every edge-vertex (that has degree $k-2$ before the addition of any vertex) one is allowed to add at most one of its two out-neighbors to ensure that its outdegree does not exceed $k-1$. Hence, an independent set of size k one-to-one corresponds to a set of vertices whose addition makes w_c the only vertex of maximum outdegree. □

In the given reduction the distinguished vertex has unbounded outdegree. Parameterized by the outdegree of the distinguished vertex MOA becomes fixed-parameter tractable.

Proposition 11.4. Max-Outdegree Addition *is fixed-parameter tractable with respect to the parameter "outdegree of the distinguished vertex w_c."*

Proof. We show that a minimum-size solution W' contains only out-neighbors of the distinguished vertex w_c. Assume that W' contains a vertex x that is not an out-neighbor of w_c. Then, deleting x from W' does not decrease the outdegree of w_c and, obviously, does not increase the outdegree of any other vertex. That is, w_c remains the vertex with maximum outdegree and W' cannot have minimum size. Hence, one can enumerate all possible subsets of $N_{\text{out}}(w_c)$ checking whether the current set forms a valid solution. Since the number of subsets of $N_{\text{out}}(w_c)$ is $2^{|N_{\text{out}}(w_c)|}$, fixed-parameter tractability follows directly. □

Concerning MIA, which is NP-hard on graphs with degree at least four, we show fixed-parameter tractability for the combined parameter "maximum indegree" and "number of added candidates."

Proposition 11.5. Min-Indegree Addition *is fixed-parameter tractable with respect to the combined parameter "maximum indegree" and "number of added candidates".*

Proof. Consider an MIA-instance. If there exists a vertex v with indegree smaller than the indegree of the distinguished vertex, one has to add at least one in-neighbor of v. Since the number of in-neighbors is bounded we can branch into all possible choices of adding an in-neighbor. In each case we can decrease parameter k by one. With the same argument as in the proof of Proposition 11.3 this leads to the running time $d_{\text{in}}^k \cdot |W|^{O(1)}$, where d_{in} denotes the maximum indegree. □

The W[2]-hardness proof for MOA/MIA restricted to tournaments is less involved than the proof for MOD/MID. It can be achieved by using the basic idea of Theorem 11.3 combined with a similar but less complicated construction of dummy candidates as the one that is used in the reduction for Theorem 11.2.

Theorem 11.5. Max-Outdegree Addition *and* Min-Indegree Addition *are W[2]-hard with respect to the parameter "number of deleted vertices" even in the case that the input graph is a tournament.*

11.2.3 W[2]-membership

We conclude the considerations on the four digraph problems by showing their containment in W[2]. We use an alternative characterization of W[2], called W*[2], introduced by Downey and Fellows [75] who showed that W*[2]=W[2]. To explain this concept, we need some definitions in the context of Boolean circuits. We distinguish two types of gates: *Large gates* are $\vee$ gates and $\wedge$ gates with unrestricted fan-in. *Small gates* are $\neg$ gates, $\vee$ gates, and $\wedge$ gates with bounded fan-in. In the "traditional" characterization of W[2] the fan-in of a small gate has to be bounded by a constant, whereas in the characterization used here it is sufficient if the fan-in of a small gate is bounded by a function of the considered parameter. The *depth* of a circuit C is the maximum number of gates on an input-output path in C. The *weft* of a circuit C is the maximum number of large gates on an input-output path in C. The k-Weighted Circuit Satisfiability (k-WCS) problem has as input a circuit C and a positive integer k, and asks whether C has a weight-k satisfying assignment (an assignment setting the values of exactly k input gates to 1). In the proof of the following theorem, we use that a parameterized problem is in W[2] if it is reducible to k-WCS for a family of circuits $\mathcal{C}$ satisfying the following two conditions [75]:

1. The weft of any circuit $C \in \mathcal{C}$ is at most two where any gate with fan-in bounded by an arbitrary function of k is considered small.

2. The depth of any circuit is at most $h(k)$ for an arbitrary function h.

With this machinery, we can show the following theorem.

Theorem 11.6. Max-Outdegree Deletion, Max-Outdegree Addition, Min-Indegree Deletion, *and* Min-Indegree Addition *are in W[2].*

Proof. First, we show the W[2]-membership for MOD by reducing it to the special case of k-WCS fulfilling Conditions (1) and (2). Let $(D = (W, A), w_c, k)$ denote an MOD-instance. If there is a vertex $w \in W$ with $d_{\text{out}}(w) \geq d_{\text{out}}(w_c) + k$, the only possibility to solve MOD is to delete w. Thus, we can assume that no such vertex exists.

The basic idea of the reduction is analogous to the proof of [75, Theorem 1]: The input variables correspond to k copies of the vertex set $W \backslash \{w_c\}$, more specifically, there are k variables for every vertex of $W \backslash \{w_c\}$. Furthermore, the construction ensures that exactly one variable of every copy of the vertex set must be set to true, that is, one "chooses" exactly k vertices (one of each copy) for the solution. Roughly speaking, this construction is useful since it enables us to "select" a subset of the chosen vertices by selecting a subset of the copies of the vertex set which are 2^k possibilities (a function only depending on k) instead of selecting a subset of at most k vertices out of W (whose size may not be bounded by a function of k).

Formally, the set of variables is $X := \{c[i, w] \mid 1 \leq i \leq k, w \in W \backslash \{w_c\}\}$. Herein, $c[i, w] = 1$ means that w is the selected vertex of the ith copy of the vertex set. Furthermore, for an integer $r \leq k$, let $S(k, r)$ denote the set of all r-element subsets of $\{1, \ldots, k\}$.

The deletion of vertices from D can affect the outdegree of w_c. In the following formulation, we try all possible amounts by which the outdegree of w_c can be decreased. Let the number by which the outdegree of w_c is decreased be s. Then, for every

other vertex $w \in W \backslash \{w_c\}$ one has that the outdegree of w must be decreased at least by $u_{w,s} = d_{\text{out}}(w) - d_{\text{out}}(w_c) - s + 1$. Note that since we assume $d_{\text{out}}(w) < d_{\text{out}}(w_c) + k$ it holds that $u_{w,s}$ is at most k. With these definitions, we can formulate a family of circuits as follows:

$$\bigvee_{s \in \{0,\ldots,k\}} \quad ((\bigvee_{J \in S(k,k-s)} \bigwedge_{j \in J} \bigvee_{w \notin N_{\text{out}}(w_c)} c[j,w]) \wedge \tag{1a}$$

$$(\bigwedge_{w \in W \backslash \{w_c\}} ((\bigvee_{j \in \{1,\ldots,k\}} c[j,w]) \vee \tag{1b}$$

$$(\bigvee_{J \in S[k,u_{w,s}]} \bigwedge_{j \in J} \bigvee_{w' \in N_{\text{out}}(w)} c[j,w']))) \tag{1c}$$

$$\wedge \quad (\bigwedge_{i \in \{1,\ldots,k\}} \bigwedge_{w \neq w'} (\neg c[i,w] \vee \neg c[i,w'])) \tag{2}$$

$$\wedge \quad (\bigwedge_{w \in W \backslash \{w_c\}} \bigwedge_{i \neq j} (\neg c[i,w] \vee \neg c[j,w])) \tag{3}$$

First, we argue that the circuits work correctly. The gates of (2) ensure that at most one vertex of every copy of the vertex set is selected and the gates of (3) ensure that every vertex is selected in at most one copy of the vertex set. The first part of the gates checks whether there is a solution for any possible outdegree which w_c can have after deleting the vertices. For this, recall that s is the amount by which the outdegree of w_c is decreased: In (1a) "the expression" becomes true if there is a size-$(k-s)$-subset of indices such that all vertices that are selected for the corresponding indices are not in $N_{\text{out}}(w_s)$. Hence, $k - s$ of the deleted vertices are not out-neighbors of w_c and, thus, the outdegree of w_c after deleting the k vertices (for which $c[i,v]$ is true) is at least $d_{\text{out}}(w_c) - s$. The gates of (1b) and (1c) ensure that for every vertex $w \in W \backslash \{w_c\}$ either w is deleted or its outdegree in the resulting instance is smaller than the final outdegree of w. More precisely, there must be either an index j for which $c[j,w]$ is true (1b) or there must be a subset of indices of size $u_{w,s}$ such that the corresponding deleted vertices are out-neighbors of w (1c). Hence, if there is a weight-k satisfying assignment, then after deleting the set of vertices $\{w \in W \mid \exists j \in \{1,\ldots,k\} \text{ with } c[j,w] = 1\}$ vertex w_c has the maximum outdegree in D.

Second, we consider the size of the circuits. Regarding the weft, the only gates with unbounded fan-in are the $\bigwedge$-gates over all vertices $w \in W \backslash \{w_c\}$ and the $\bigvee$-gates over the out-neighbors of a vertex. It is easy to check that there are at most two gates of this type at one input-output path. The depth of the circuit is obviously bounded by a constant. Thus, MOD is contained in W[2].

For the other three problems one can use the same methods to show the membership in W[2]. For the vertex addition problems the variable set contains only copies of all vertices that can be added. Then, for MOA, MID, and MIA one can adapt the first part of the constraints in a straightforward manner. □

11.3 Parameterized complexity of candidate control

In this section, we come back to voting systems. The digraph problems considered in the previous section turn out to be very useful to determine the parameterized complexity of candidate control for different voting systems. In Subsection 11.3.1, we briefly discuss some consequences of the results obtained for the digraph problems for control in Llull and Copeland voting. In Subsection 11.3.2, we show NP-hardness for candidate control in Llull and Copeland voting for a constant number of votes. Finally, in Subsection 11.3.3, we provide parameterized intractability results with respect to the "number of deleted/added candidates" for plurality and Copeland$^{\alpha}$ voting.

As in Part II, the *position* of a candidate b in a vote v is the number of candidates that are better than b in v plus one. That is, the leftmost (and best) candidate in v

has position 1 and the rightmost has position m.

11.3.1 Llull and Copeland voting

The only difference between Llull and Copland voting is the way in which ties are evaluated. If two candidates are tied in their head-to-head contest, both of them get zero points in a Copeland election and one point in a Llull election. As stated by Faliszewski et al. [97], the different evaluation of ties can make the dynamics of Llull's system quite different from those of Copeland's system. For example, they observed that the proof techniques used to show NP-hardness are quite different for different ways of evaluating ties. However, for the problems considered in this work, until now, there was no measurable difference in the computational complexity of candidate control between Llull and Copeland voting. Using a multivariate view on the problems, we identify cases in which their complexities differ. As CC-DC-Llull/CC-AC-Llull and CC-DC-Copeland/CC-AC-Copeland are FPT-equivalent to MID/MIA and MOD/MOA, respectively, all results of Tables 11.1 and 11.3 also hold for them. In particular, the bounded-degree scenario for MID (Proposition 11.2) seems to be fairly realistic: To control an election is particularly attractive if the distinguished candidate is already "close" to be a winner. A natural indicator for "closeness" is the number of candidates that beat the distinguished candidate. That is, the corresponding distinguished vertex has bounded indegree. In this case, in contrast to Copeland elections, Llull elections are "easy" to control. Thus, as a direct consequence of Theorem 11.1 and Proposition 11.2, we obtain the following.

Corollary 11.1. *CC-DC-Llull is fixed-parameter tractable with respect to the parameter "number of candidates defeating the distinguished candidate". CC-DC-Copeland is NP-hard to control even if for every candidate the number of candidates that are not tied with it is at most three.*

11.3.2 Number of votes as parameter

In many election scenarios there is only a small number of votes. For example, consider a human resources department where few people are deciding which job applicant gets the employment. Another prominent example is rank aggregation. An open question of Faliszewski et al. [98] regards the parameterized complexity of candidate control in Copeland$^{\alpha}$ voting with respect to the parameter "number of votes". We answer this question for the two important special cases Llull and Copeland by making use of the corresponding digraph problems. More precisely, we devise four reductions showing that the problems of controlling Llull and Copeland voting by deleting or adding candidates are NP-complete even in the case of a constant number of votes. Each reduction is from a special case of the corresponding digraph problem. For all but one reduction the NP-hardness of the considered special case follows from reductions given in Section 11.2. The new part of the reductions is to show how a given instance of the digraph problem can be encoded into an election using a constant number of votes. Recall that, as discussed in Section 11.1, we say that a digraph encodes an election if the outcomes of the pairwise head-to-head contests reflect the arcs in the digraphs. That is, if there is an arc from vertex v to vertex w, then the corresponding candidate v must be better than w in more than half of the votes. Here, the encoding

of all digraphs into a constant number of votes is based on the idea to partition the set of arcs into a constant number of subsets in a way such that each subset can be encoded independently of the others by each time two votes.[4] A useful tool to obtain such partitionings are arc colorings for digraphs.

Lemma 11.1. *If there is a proper ℓ-arc coloring for a digraph D, then D can be encoded into 2ℓ votes.*

Proof. Given a digraph $D = (V, A)$ and a corresponding proper ℓ-arc coloring $\mathcal{C} : A \to \{\mathcal{R}_1, \mathcal{R}_1, \ldots, \mathcal{R}_\ell\}$ for D. In the underlying undirected graph of D the edges of the same color class form a matching, that is, two arcs of the same color do not share a common vertex. Hence, the coloring $\mathcal{C}$ partitions the arc set into ℓ classes of independent arcs. We next describe how the arcs of graph D can be encoded in an election with 2ℓ votes. Let $A_{\mathcal{R}_1} = \{(r_1, r_{1'}), \ldots, (r_q, r_{q'})\}$ denote the arcs colored by $\mathcal{R}_1$. Furthermore, $W_{\overline{\mathcal{R}_1}}$ denotes the set of vertices that are not incident to any arc of $A_{\mathcal{R}_1}$. To encode $A_{\mathcal{R}_1}$, we add the two votes

$$v_{\mathcal{R}_1,1} : r_1 > r'_1 > r_2 > r'_2 > \cdots > r_q > r'_q > W_{\overline{\mathcal{R}_1}}$$

$$v_{\mathcal{R}_1,2} : \overline{W_{\overline{\mathcal{R}_1}}} > r_q > r'_q > \cdots > r_2 > r'_2 > r_1 > r'_1$$

to the election. In the same way, for each $1 < i \leq \ell$ we add two votes $v_{\mathcal{R}_i,1}$ and $v_{R_i,2}$ for the arcs colored by $\mathcal{R}_i$. The correctness of the construction follows from two observations. First, since the arcs of the same color do not share common endpoints, in every vote all vertices occur exactly once and we have a valid election. Second, consider an arc $(w', w'') \in A$ with $\mathcal{C}((w', w'')) = R_i$ for some $1 \leq i \leq \ell$. Then, w' defeats w'' in the votes $v_{\mathcal{R}_i,1}$ and $v_{\mathcal{R}_i,2}$ and ties with w'' in the remaining votes. Moreover, since every arc occurs in exactly one color class, all arcs are encoded, and, since all other candidates are tied in all pairs of the votes, we have ties between all other pairs of candidates. □

Every undirected graph admits a proper arc/edge-coloring using $\Delta + 1$ colors, where Δ denotes the maximum degree. Moreover, Δ is a lower bound on the number of colors that are necessary for any proper arc/edge coloring. For arbitrary graphs, it is NP-complete to decide whether the graph has an proper Δ-arc/edge coloring. In contrast, by König's Theorem, for all bipartite graphs one can find a proper Δ-arc/edge coloring in polynomial time [139].

Lemma 11.2. *(König [1916]) A bipartite graph is Δ-edge-colorable, where Δ denotes the maximum degree of the graph. A corresponding proper Δ-edge coloring can be computed in polynomial time.*

These two lemmas are used to show the following.

Theorem 11.7. *Controlling Llull and Copeland by deleting/adding candidates is NP-complete for a constant number of votes. More precisely,* CC-DC-Copeland *is NP-complete for six votes,* CC-AC-Copeland *is NP-complete for eight votes,* CC-DC-Llull *is NP-complete for ten votes, and* CC-AC-Llull *is NP-complete for eight votes.*

[4] In contrast, in previous works, as for example [97], only one arc was encoded into two votes.

Proof. For all problems NP-membership is obvious. We start with the NP-hardness proof for CC-DC-Copeland to demonstrate the basic idea. Consider the reduction from the NP-complete 3X-2-Hitting Set to MOD as depicted in Fig. 11.1. The digraph D of a resulting MOD-instance (D, w_c, k) has maximum degree three and the underlying undirected graph of D is bipartite. More precisely, one partition consists of the subset-vertices and w_c, and the other partition consists of the element-vertices and the neighbors of w_c. Note that as we reduce from 3X-2-Hitting Set, we do not have any further dummy vertices. It follows directly from Lemma 11.2 that D has a proper 3-arc coloring. Thus, by Lemma 11.1, D can be encoded into an election of six votes resulting in an equivalent instance of CC-DC-Copeland.

Next, we argue that CC-AC-Llull is NP-complete for eight votes. According to Theorem 11.3 MIA is NP-complete even when restricted to graphs with maximum degree four. Moreover, observe that the underlying undirected graph of the digraph constructed in the respective reduction from 3X-2-Hitting Set (see Fig. 11.3) is bipartite. Hence, for CC-AC-Llull the NP-hardness follows in complete analogy to CC-DC-Copeland by using Lemmas 11.1 and 11.2.

For CC-AC-Copeland, we show how to encode an NP-hard MOA-instance that results from the reduction of $3d-$Independent Set (Fig. 11.4) into an election of eight votes. Note that since MOA is fixed-parameter tractable with respect to the maximum degree, it is polynomial-time solvable for an instance with constant degree. Hence, we can not assume that the maximum degree in the constructed MOA-instance is constant. However, by using the reduction from $3d$-Independent Set ($3d$-IS) we can still make use of a degree restriction of the subgraph induced by $\{e_i \mid i = 1, \ldots, m\} \cup \{u_j \mid j = 1, \ldots, n\}$. Since the degree within this subgraph is at most three and its underlying undirected graph is bipartite, due to Lemmas 11.1 and 11.2 it can be encoded into six votes. The remaining arcs can be encoded into two further votes as follows. Let $S(e_i)$ denote the $k-2$ dummy out-neighbors of e_i, then we can add the following two votes

$e_1 > S(e_1) > \cdots > e_m > S(e_m) > w_c > u_1 > \cdots > \underline{u_n > x} > y_1 > \cdots > \underline{y_{k-1}}$

$x > y_1 > \cdots > y_{k-1} > w_c > u_n > \cdots > u_1 > e_m > \overline{S(e_m)} > \cdots > e_1 > \overline{S(e_1)}$.

This completes the proof for CC-AC-Copeland.

Finally, we show that CC-DC-Llull is NP-hard for ten votes. We present a reduction from $3d$-IS to MID and show that the resulting MID-instance can be encoded into an election with ten votes. Note that since CC-DC-Llull is solvable in polynomial time on acyclic digraphs and FPT with respect to the degree, in contrast to the other problems, there is no previous reduction we can reuse.

Given a $3d$-IS-instance consisting of an undirected graph $G = (U, E)$ with $n := |U|$ and a non-negative integer k, we construct an MID-instance consisting of a graph $D = (W, A)$, a distinguished vertex w_c, and a non-negative integer k. See Fig. 11.5 for an illustration. The vertex set W is the disjoint union of the following sets:

- $\{w_c\}$, the distinguished vertex,
- $U' := \{u'_i \mid u_i \in U\}$, one new vertex for every IS-vertex,
- $E' := \{e'_i \mid e_i \in E\}$, one new vertex for every IS-edge,
- three sets of dummy vertices $X := \{x_1, x_2, \ldots, x_n\}$, $Y := \{y_1, y_2, \ldots, y_n\}$, and $Z := \{z_1, z_2, \ldots, z_n\}$ that are needed to "set" the indegrees of the other vertices in an appropriate way, and

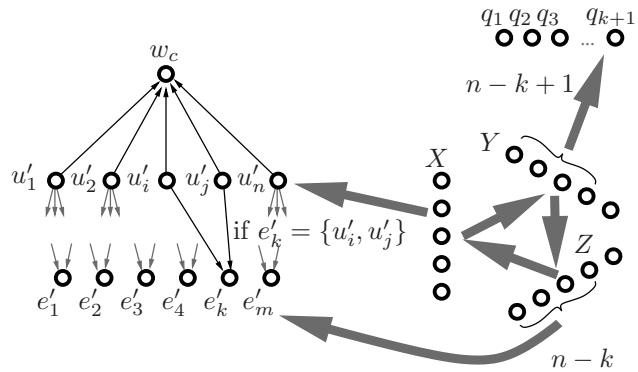


Figure 11.5: Reduction from an $3d$-Independent Set-instance (left) to an Min-Indegree Deletion-instance (right).

- $Q := \{q_1, q_2, \ldots, q_{k+1}\}$, vertices that enforce that the indegree of the distinguished vertex must decrease by k.

The basic idea is to set the arcs such that the indegree of the distinguished vertex w_c has to be decreased by k. Furthermore, to decrease the indegree of w_c one can only delete vertices that correspond to vertices of the $3d$-IS-instance, that is, vertices of U'. The deletion of such a vertex does not only decrease the indegree of w_c but also the indegree of the (at most three) vertices that correspond to its incident edges. By using the dummy vertices one can set the indegrees of the edge-vertices such that one can delete at most one neighbor of any edge-vertex. Then, to make w_c a winner one has to delete k vertices that correspond to vertices of the $3d$-IS-instance such that for every edge at most one of its incident vertices is deleted. Thus, the deleted vertices must correspond to an independent set. In the following, we describe the arc set A that is given by the union of the following disjoint arc sets.

- $A_{U',w_c} := U' \times \{w_c\}$,
- $A_{U',E'} := \bigcup_{u_j \in U}(\{u'_j\} \times \{e'_i \mid e_i \in E \wedge u_j \cap e_i \neq \emptyset\})$,
- $A_{X,Y,Z} := (X \times Y) \cup (Y \times Z) \cup (Z \times X)$,
- $A_{X,U'} := X \times U'$,
- $A_{Y,Q} := \{y_1, y_2, \ldots, y_{n-k+1}\} \times Q$, and
- $A_{Z,E'} := \{z_1, z_2, \ldots, z_{n-k}\} \times E'$.

For an illustration of the construction see Fig. 11.5. Note that every vertex e'_i has exactly two in-going arcs from U' and $n - k$ from Z. Hence, it can easily be verified that in D the indegree for all $e'_i \in E'$ is $n - k + 2$, the indegree of q_i is $n - k + 1$ for $i = 1, \ldots, k$, and the indegree of all remaining vertices is n.

We next prove the correctness of the reduction. Let $I \subseteq U$ be an independent set of G. After deleting $S := \{u'_i \mid u_i \in I\}$ from D we have $d_{\text{in}}(w_c) = n - k$ and since I is an independent set the indegree of every vertex $e'_i \in E'$ is decreased by at most

one, that is, $d_{\text{in}}(e'_i) \geq n - k + 1$. The indegree of all other vertices is not affected. Therefore, w_c is the vertex with minimum indegree.

Let $S \subseteq W$ be an optimal solution for MID. We can assume that $S \subseteq U'$, since in order to improve w_c against the vertices from Q we must delete k vertices of $N_{\text{in}}(w_c)$. This is due to the fact that we cannot delete all $k+1$ vertices from Q. Since S contains only vertices from U', the indegree of w_c is exactly $n - k$. Moreover, for every $e_i = \{u_j, u_k\} \in E$ in order to ensure that $d_{\text{in}}(e'_i) > n - k$ we can have either u'_j or u'_k in the solution. Hence, $\{u_i \in U \mid u'_i \in S\}$ is an independent set.

In the remainder of this proof we show that the graph D can be encoded into an election using ten votes in total. Because G has maximum degree 3, it is easy to see that the underlying undirected graph of $D[U' \cup E']$ is bipartite and has maximum degree three. Consequently, following Lemma 11.2, there exists a proper 3-arc coloring for $D[U' \cup E']$ and the information for this subgraph can be encoded into six votes (Lemma 11.1). Let $R := A \backslash \{X, Y, Z\}$. The arcs between X, Y, and Z can be encoded into the following three pairs of votes.

1. $X > Y > Z > R$ and $\overline{R} > \overline{X} > \overline{Y} > \overline{Z}$.
2. $Y > Z > X > R$ and $\overline{R} > \overline{Y} > \overline{Z} > \overline{X}$.
3. $Z > X > Y > R$ and $\overline{R} > \overline{Z} > \overline{X} > \overline{Y}$.

Since the arcs between X, Y, and Z are independent from the arcs in $D[U' \cup E']$, the encoding of both sets of arcs can be done within the same three pairs of votes. It remains to encode the arcs from Y to Q, the arcs from X to U', the arcs from Z to E', and the arcs from U' to w_c. It is not hard to see that this can be done by using two further pairs of votes. □

11.3.3 Number of deleted/added candidates as parameter

To control an election without raising suspicion one may add or delete only a limited number of candidates. Here, we investigate whether it is possible to obtain efficient algorithms under this assumption. More specifically, we consider the parameterized complexity of destructive and constructive control by adding or deleting a fixed number of candidates. Our results are summarized in Table 11.2. It turns out that all NP-complete problems are intractable from this parameterized point of view as well. This even holds true for plurality voting, which can be considered as the "easiest" voting system in terms of winner evaluation and for which the Manipulation problem can be solved optimally by a simple greedy strategy [65]. Whereas the results for Copeland$^\alpha$ voting can be obtained easily from the results of the corresponding digraph problems, we give two reductions with new ideas for constructive and destructive control in plurality voting.

Copeland$^\alpha$

Having no ties in the pairwise head-to-head contests between all pairs of candidates is a realistic scenario. It is always the case for an odd number of votes and likely for a large number of votes. Thus, it is interesting to investigate this setting. Note that the NP-hardness proofs of candidate control in Copeland$^\alpha$ voting rely on ties [100]. For elections without ties in all pairwise head-to-head contests, CC-DC-Copeland$^\alpha$, as

well as CC-AC-Copeland$^\alpha$, coincide for all $0 \leq \alpha \leq 1$, since these problems only differ in the way ties are evaluated.

As discussed in the introduction, MOD/MOA and CC-DC-Copeland/CC-AC-Copeland are FPT-equivalent. Using the same reductions one can show that MOD/MOA in tournaments are FPT-equivalent to CC-DC-Copeland$^\alpha$/CC-AC-Copeland$^\alpha$ without ties. Thus, we obtain the following corollary from Theorem 11.2 and Theorem 11.5.

Corollary 11.2. *For a tie-free voting and $0 \leq \alpha \leq 1$, CC-DC-Copeland$^\alpha$ is W[2]-complete with respect to the "number of deleted candidates" and CC-AC-Copeland$^\alpha$ is W[2]-complete with respect to the "number of added candidates".*

Plurality

In this section, we consider plurality voting and show that candidate control is not only NP-hard but also intractable from parameterized point of view. Note that the class containment in W[1] or W[2] for all kinds of candidate control in plurality voting is open.

For plurality voting, the W[2]-hardness results for control by adding candidates follow from existing NP-hardness proofs [13, 131]. Hence, we can state the following theorem.

Theorem 11.8. *Destructive and constructive control of plurality voting by adding candidates are W[2]-hard with respect to the "number of added candidates".*

In contrast, the reductions used to show NP-hardness for destructive and constructive control by deleting candidates [13, 131] do not imply their W[1]-hardness. Thus, we develop new parameterized reductions. For the constructive case we can show W[2]-hardness by a reduction from MOD. Note that the encoding of an MOD instance into a plurality election is more demanding than for Copeland voting and the other direction (encoding a plurality election into an MOD instance) is not obvious. Therefore, in contrast to the considerations for Copeland$^\alpha$ elections, where the main focus was on showing the W[2]-hardness of MOD on tournaments, here the technical part is the reduction from MOD to CC-DC-Plurality itself.

Theorem 11.9. *Constructive control of plurality voting by deleting candidates is W[2]-hard with respect to the parameter "number of deleted candidates".*

Proof. We present a parameterized reduction from MOD. The basic idea is to construct a plurality election such that, for every vertex w of the MOD-instance with higher outdegree than the distinguished vertex w_c, the corresponding candidate w' has a higher plurality score than the distinguished candidate c. More precisely, the difference between the score of w' and the score of c equals the difference of their outdegrees, that is, $\text{score}(w') - score(c) = d_{\text{out}}(w') - d_{\text{out}}(c)$. Furthermore, due to our construction there are only two possibilities to make c to beat w' in the plurality election. First, one can delete w' itself. Second, the deletion of a candidate corresponding to an out-neighbor of w decreases the score of w' by one point but the score of c remains unchanged. Thus, in this case, one has to delete at least $d_{\text{out}}(w') - d_{\text{out}}(c) + 1$ candidates that correspond to out-neighbors of w'. In both cases the deletion of the corresponding vertices in the MOD-instance has the effect that the distinguished vertex has higher outdegree than w. In the following, we describe the formal construction.

Given an MOD instance $(D = (W, A), w_c, k)$ with $W = \{w_1, w_2, \ldots, w_n\}$ and $w_c = w_1$, we construct an election (V, C) as follows: We have one candidate corresponding to every vertex, that is, $C' := \{c_i \mid w_i \in W\}$. The set of candidates C then consists of C' and an additional set F of "dummy" candidates (only used to "fill" positions that cannot be taken by other candidates in our construction). The multiset of votes V consists of two subsets V_1 and V_2. In V_1, for every $c_i \in C'$ we have $d_{\text{out}}(w_i)$ votes in which c_i is at the first position and with dummy candidates in the positions from 2 to $k+1$. Then, for every such vote, the remaining candidates follow in arbitrary order. In V_2, for every $c_i \in C'$ we have $|W|$ votes in which c_i is at the first position. For all candidates $c_j \neq c_i$ with $w_j \notin N_{\text{in}}(w_i)$, we ensure that in exactly one of these $|W|$ votes c_j is at the second position. In all other of these votes, the second position is filled with a dummy candidate. Moreover, we add dummies to all positions from 3 to $k+1$. Concerning the dummies, in V_1 and V_2 we ensure that every dummy candidate $f \in F$ has a position better than $k+2$ only in one of the votes. This can be done by using a different dummy candidate for every position. Obviously the size of F is less than $(k+1) \cdot |V|$. The dummies exclude the possibility of "accidently" getting candidates in the first position. Note that by deleting k candidates only a candidate that is at one of the first $k+1$ positions in a vote has the possibility to increase its plurality score. Furthermore, by construction, the dummy candidates fulfill the following two conditions. First, the score of a dummy candidate can become at most one. Second, it does never make sense to delete a dummy as by this only other dummies can get into the first position of a vote. Next, we prove the correctness of the reduction.

Claim: Candidate c_1 can become the plurality winner of (V, C) by deleting k candidates iff w_1 can become the only maximum-degree vertex in D by deleting k vertices.

"$\Rightarrow$": Denote the set of deleted candidates by R. We show that after deleting the set of vertices $W_R := \{w_i \mid c_i \in R\}$ the vertex w_1 is the only vertex with maximum degree. Before deleting any candidates, for every candidate c_i we have $\text{score}(c_i) = \text{score}(c_1) + s_i$ with $s_i := d_{\text{out}}(w_i) - d_{\text{out}}(w_1)$. After deleting the candidates in R, candidate c_1 is the winner. Hence, for $i = 2, \ldots, |W|$ we must have either that $\text{score}(c_i) < \text{score}(c_1)$ or that c_i is deleted. For a non-deleted candidate c_i with $i > 1$ the difference between $\text{score}(c_i)$ and $\text{score}(c_1)$ must be decreased by at least $s_i + 1$. By construction, the only way to decrease the difference by one is to delete a candidate such that c_1 becomes first in one more vote and c_i does not increase the number of its first positions. All candidates that can be deleted to achieve this correspond to vertices in $N_{\text{in}}(w_i) \setminus N_{\text{in}}(w_1)$. To improve upon c_i we must delete at least $s_i + 1$ candidates that fulfill this requirement. Hence, in D the outdegree of w_i is reduced to be less than the outdegree of w_1.

"$\Leftarrow$": Let $T \subseteq D$ denote the solution for MOD. We can show in a straightforward way ("reverse" to the other direction) that by deleting the set of candidates $C_T := \{c_i \mid w_i \in T\}$ candidate c_1 becomes a plurality winner. □

In contrast to Copeland$^\alpha$ voting, for plurality voting destructive control by deleting candidates is NP-hard [131]. We show that it is even W[1]-hard by presenting a parameterized reduction from the W[1]-complete Clique problem [76]. Given an undirected graph $G = (W, E)$ and a positive integer k, the Clique problem asks to decide whether G contains a complete subgraph of size at least k.

Theorem 11.10. *Destructive control of plurality voting by deleting candidates is W[1]-hard with respect to the parameter "number of deleted candidates".*

Proof. Given a Clique instance $(G = (W, E), k)$, we construct an election as follows. The set of candidates is

$$C := C_W \uplus C_E \uplus \{c, w\} \uplus D$$

with $C_W := \{c_u \mid u \in W\}$, $C_E := \{c_{uv} \mid \{u, v\} \in E\}$, and a set of dummy candidates D. In the following, the candidates in C_W and C_E are called *vertex candidates* and *edge candidates*, respectively. Furthermore, we construct the votes in a way such that w is the candidate that we would like to prevent from winning, c is the only candidate that can beat w, and D contains dummy candidates that can gain a score of at most one. In the multiset of votes V we have for every vertex $u \in W$ and for each incident edge $\{u, v\} \in E$ one vote of the type $c_u > c_{uv} > c > \ldots$, that is, there are $2 \cdot |E|$ votes of this type, two for every edge. Additionally, V contains $|W| + k \cdot (k-1)$ votes in which w is at the first position and $|W| + 1$ votes in which c is at the first position. That is, the score of w exceeds the score of c by $k \cdot (k-1)$. In all votes, the remaining free positions between 2 and $k + \binom{k}{2} + 1$ are filled with dummies such that every dummy occurs in at most one vote at a position better than $k + \binom{k}{2} + 2$. This can be done using less than $|V| \cdot (k + \binom{k}{2} + 1)$ dummy candidates. In every vote the candidates that do not occur in this vote at a position less than $(k + \binom{k}{2} + 1)$ follow in arbitrary order.

Claim: Graph G contains a clique K of size k if and only if candidate c can become plurality winner by deleting $k' := k + \binom{k}{2}$ candidates.

"$\Rightarrow$": Delete the $k + \binom{k}{2}$ candidates that correspond to the vertices and edges of K. Then, for every of the $\binom{k}{2}$ deleted edge candidates we also deleted the two vertex candidates that correspond to the endpoints of the edge. Therefore, for every of the $\binom{k}{2}$ edges candidate c gets in the first position in two more votes. Hence, the score of candidate c is increased by $2 \cdot \binom{k}{2} = k \cdot (k-1)$ and the score of candidate w is not affected. Thus, the total score of w is $|W| + k \cdot (k-1)$ and the total score of c is $|W| + k \cdot (k-1) + 1$ and w is defeated by c.

"$\Leftarrow$": By construction, we cannot decrease the score of w and we cannot increase the score of a vertex candidate (which is at most $|W| - 1$). Furthermore, by the deletion of at most k' candidates the score of a dummy candidate can become at most one, and the score of an edge candidate can become at most two. Hence, c is the only candidate that can prevent w from winning. Furthermore, as the deletion of at most k' dummies never moves c into a first position, we can assume that the solution deletes only edge and vertex candidates. Thus, it remains to that the only way to increase the score of c by at least $k \cdot (k-1)$ is to choose edge and vertex candidates that correspond to the vertices and edges of a clique of size k.

Let $C_{W'} \cup C_{E'}$ be a solution of size k', that is, deleting the candidates in $C_{W'} \cup C_{E'}$ prevents candidate c from winning. Let W' and E' be the corresponding vertices and edges and let $i := |W'|$. It is easy to see that $i \leq k$ since the deletion of an edge candidate moves c in exactly two votes from the third to the second position. Hence, in order to move c in at least $k \cdot (k-1)$ votes to the first position, we have to delete at least $(k \cdot (k-1))/2 = \binom{k}{2}$ edge candidates. Consequently, since $k' = \binom{k}{2} + k$, we can delete at most k further vertex candidates.

In the following, we show that we have to remove exactly k vertex candidates, that is, we must have $i = k$. Consider the election after deleting the candidates in $C_{W'}$. Let $E'_1 \subseteq E'$ be the set of edges with both endpoints in W' and let $E'_2 := E' \setminus E'_1$.

Clearly, by deleting $C_{W'} \cup C_{E'}$ the score of c increases by at most $2 \cdot |E'_1| + |E'_2|$. Since $C_{W'} \cup C_{E'}$ is a solution, we obtain

$$2 \cdot |E'_1| + |E'_2| \geq \text{score}(c) - |W| - 1 \geq k \cdot (k-1). \tag{11.1}$$

Furthermore, we know that

$$|E'_1| + |E'_2| + i \leq k + \binom{k}{2}. \tag{11.2}$$

Inequality (11.1) implies that the score of c becomes maximum if E_1 is maximum, that is, if the graph (V', E'_1) is complete. In this case, E'_1 has cardinality $\binom{i}{2}$ and, hence, according to Inequality (11.2) the candidate set E_2 has cardinality at most $k' - \binom{i}{2} - i$ and the score of c is at most $2 \cdot \binom{i}{2} + k' - \binom{i}{2} - i + |W| + 1$. Assume that we have $i < k$, then $\text{score}(c) - |W| - 1 \leq 2 \cdot \binom{i}{2} + k' - \binom{i}{2} - i < k \cdot (k-1) = \text{score}(c) - |W| - 1$, a contradiction. Hence, we must have that $i = k$ and $|E'| = |E'_1| + |E'_2| = \binom{k}{2} = (k \cdot (k-1))/2$. Therefore, $E'_2 = \emptyset$ and (W', E') is a clique. □

11.4 Conclusion

In this chapter, we investigated the parameterized complexity of four new digraph modification problems and of electoral candidate control. Somewhat surprisingly, the problems turned out to be intractable in almost all settings. For instance, Max-Outdegree Deletion is W[2]-complete for two very restricted digraph classes, tournaments and acyclic graphs. We conclude with several remarks and concrete questions regarding future research.

- The investigation of the parameterized complexity of voting systems other than plurality and Copeland$^\alpha$ seems to be of interest. For example, Erdélyi et al. considered electoral control for "sincere-strategy preference-based approval voting (SP-AV)"[88] and fallback voting [90, 91] regarding its classical complexity. Whereas for fallback voting there is a first study of their parameterized complexity of control [89], for SP-AV the parameterized complexity has not been considered yet.

- Regarding candidate control in plurality voting, we gave W[1]-/W[2]-hardness results with respect to the number of added/deleted candidates. The class containment was left open. Note that polynomial-time solvability for constant parameter values, that is, containment in XP, is obvious.

- We only considered the parameterized complexity for candidate control. There are many other settings to study, for example, control by adding or deleting votes. This direction of research can be further extended by investigating multi-mode attacks as suggested by Faliszewski et al. [95]. Note that their resistance results exploit that for every kind of attack one can specify how often it may be applied. For example, can a distinguished candidate become a winner by deleting k candidates and adding k' votes? It is an interesting open question whether resistance transfers to the setting that only the total number of operations is bounded.

- We have shown that constructive candidate control for Llull and Copeland voting is NP-hard for a constant number of votes, answering an open question from Faliszewski et al. [99] for these two special cases of Copeland$^\alpha$. For Copeland$^\alpha$ voting with $0 < \alpha < 1$, for both kinds of candidate control the parameterized complexity with respect to the number of votes is still open. Besides developing a method that works for general α, an interesting first step would be to try to extend our results to other interesting special cases like $\alpha = 0.5$.

- In contrast to manipulation [64, 116, 179, 204], to the best of our knowledge all investigations for control focused on worst-case scenarios. There seem to be no studies that are concerned with strategies that allow for efficient control in the average case or for "most" instances. Considerations in this direction seem to be of interest.

- Liu et al. [157] studied the parameterized complexity of some settings of control for plurality, Condorcet, and approval. One of their main results is to show W[2]-hardness for constructive and destructive control by adding candidates for plurality. However, in the conference version of our paper [29] (which is cited by Liu et al. [157]) we explicitly stated that these results directly follow from the corresponding NP-hardness reductions in literature (see Table 11.2). They also provided W[1]-hardness for control by deleting votes for Condorcet voting. This problem is clearly the same as Young Score for which we provided W[2]-completeness in a former work (see Section 6.2).

Whereas the above points are all concerned with the voting problems, also the introduced digraph problems led to further research topics. In recent work [17] (see also [45]), we obtained some results for several parameterization measuring the "distance from acycility" for Min-Indegree Deletion and further related problems. The starting point for these studies has been the observation that MID on acyclic graphs is solvable in polynomial time (Proposition 11.2).

Chapter 12

Summary and future research directions

This thesis investigated the use of a multivariate complexity analysis to better understand the computational complexity of voting problems and to develop efficient algorithms capturing relevant scenarios. In the following two sections, we first summarize our results and then discuss some directions for future research.

12.1 Summary of results

The overall goal of this thesis is to contribute to a systematic analysis of the multivariate complexity of voting problems. In particular, the thesis promotes a more fine-grained study of the computational complexity by pursuing parameterized algorithmics. In the following, we summarize our concrete results.

In Part I of the thesis, we proposed fixed-parameter algorithms as a possible way to compute winners for in general NP-hard-to-evaluate voting rules. For the important Kemeny voting rule, we devised a systematic parameterized complexity analysis with respect to several parameterizations. Herein, our contributions can be described according to the following steps:

1. We identified meaningful parameterizations such as the "average KT-distance" or the "maximum range of a candidate".

2. We identified intractable cases and designed several fixed-parameter algorithms by applying a variety of algorithmic techniques comprising depth-bounded search trees, data reduction rules, and dynamic programming.

3. The implementation of some of the algorithms led to a freely available software tool and experimental evaluations with real-world data showed the practical use of some of our algorithms.

In addition to a comprehensive study of Kemeny's rule, we investigated the parameterized complexity of winner determination in Dodgson and Young elections. In

particular, we presented a fixed-parameter algorithm for Dodgson Score answering an open question of Christian et al. [56].

While Part I contains results for individual voting rules, in Part II our first result comprises the classification of the computational complexity of the Possible Winner problem for the natural class of pure scoring rules. By devising an appropriate framework and providing intricate many-one reductions we almost arrived at a dichotomy. The computational complexity of the only remaining case was settled by Baumeister and Rothe [14]. Motivated by the NP-hardness of Possible Winner for nearly all pure scoring rules, we provided a parameterized complexity analysis. For some parameters, we established fixed-parameter tractability for all scoring rules. In contrast, we showed NP-hardness for a constant number of votes for Borda and k-approval. This led the way to a parameterized complexity analysis for Possible Winner under k-approval with respect to the combined parameter k and number of votes. The corresponding kernelization and non-existence of a polynomial kernel results are among the first such results for voting problems.

In Part III, we extended previous studies of candidate control in elections to the realistic scenario of deleting or adding only a bounded number of candidates. In the course of this study, we partially answered an open question of Faliszewski et al. [98] by providing NP-hardness results for a constant number of votes for two special cases of Copeland$^{\alpha}$ elections.

12.2 Future challenges

For open questions and remarks concerning the specific problems considered in this thesis, we refer to the corresponding chapters. More specifically, questions regarding Kemeny Score can be found in Sections 3.9, 4.4, and 5.3, regarding Dodgson Score and Young Score in Section 6.3, regarding Possible Winner mainly in Chapter 10 and also in Section 9.3, regarding candidate control in Section 11.4. In the following, we discuss more general issues distinguishing between problem-oriented, technique-oriented, and parameter-oriented approaches.

12.2.1 Problem-oriented approaches

The approach used in this work, and maybe the most obvious approach to start a systematic multivariate complexity analysis of a field, is to consider the complexity of specific problems. As discussed in Section 2.3, up to now only few works have provided such studies. Hence, there are many problems in computational social choice to be explored in future studies. In many cases, there are some meaningful obvious parameterizations, for example, the number of deleted candidates when controlling an election. Some systematic ways to identify further meaningful parameterizations are described by Niedermeier [172]. We briefly sketch two of them and then discuss property-oriented approaches generalizing problem-oriented ones.

Identifying parameters. A useful method to identify further parameters might be to *deconstruct intractability* [147, 172]. Herein, the basic idea is to investigate the NP-hardness proof of the studied problem and find out why the constructed instances might not reflect practically relevant settings. We illustrate this for the NP-hardness

proof of Kemeny Score for four votes [77]. In the corresponding construction only for few candidate pairs the pairwise head-to-head contest are not tied. As a consequence, the constructed instances come with a large average KT-distance and do not capture many realistic settings (see Chapter 3).

An approach that should be pursued whenever one is interested in solving problems on real-world data regards the identification of parameterizations based on data analysis. The basic idea is to experimentally measure properties of the input to find out which of them come with small parameter values. This leads the way to a "data-driven" algorithm design followed by algorithm engineering exploiting small parameter values.

Property-oriented approaches. A problem-oriented approach naturally generalize to a *property-oriented* approach obtaining results for whole classes of voting rules. An example is provided by Elkind et al. [82]. They showed a fixed-parameter tractability result for classes of voting rules defined according to "distance rationalizability" (see also Section 6.3). One particularly interesting line of research in this direction might be to investigate how some of the "standard" properties of voting rules can help in algorithm design. For example, Pini et al. [178] used "Independence of Irrelevant Alternatives" to obtain polynomial-time algorithms for the Possible Winner problem (see also Chapter 10). Another question that might come into mind immediately is whether "consistency" can help to design divide-and-conquer algorithms for winner determination. Such investigations also might lead to interesting parameterizations, for example, one might measure the "distance from a desired property" according to an appropriate distance measure (as done in [82]).

12.2.2 Technique-oriented approaches

We highlight some issues concerned with algorithmic techniques useful to design fixed-parameter algorithms.

Kernelization. A seemingly difficult task in the development of data reduction rules for voting problems concerns the design of data reduction rules "reducing" the number of votes. This includes the design of general reduction rules, for example, replacing a multiset of votes by a "bounded-size" multiset such that the outcomes of the pairwise head-to-head contests between the candidates remain unchanged. Although such a data reduction rule would be a main ingredient for obtaining kernelization results for voting problems in general, we are not aware of any such data reduction rule even only working for a specific voting rule.

In this work, we "circumvented" this problem by two ways. First, in Chapter 4, we introduced the concept of partial kernels not necessarily bounding the number of votes but still coming with a provable performance guarantee. Second, the kernelization results obtained for Possible Winner for k-approval in Chapter 9 are based on combined parameters where the number of votes is part of the parameter. Similarly, the simple kernelization for Kemeny Score with respect to the parameter Kemeny score (see Section 3.4) relies on the fact that the parameter assumes "large" values for non-trivial instances. In this sense, instead of applying data reduction rules that cut away parts of the votes, the argumentation works in the following way. The number

of votes in a "non-trivial" yes-instance must be already bounded by the parameter, otherwise one can conclude that it is a no-instance.

Dynamic programming based on deficit vectors. The basic idea of the dynamic programming algorithm devised to show fixed-parameter tractability for Dodgson Score in Section 6.1 is to investigate "deficit vectors" that are bounded by the parameter. The same idea has also turned out useful to design a fixed-parameter algorithm for Possible Winner for k-approval with respect to k for constant number of zero-positions (see Subsection 9.1.2). In addition, we could use the idea to design a dynamic programming algorithm in a recent work [17] (see also [45]). This indicates that this kind of dynamic programming might be of general interest (see also the following subsection). However, a serious drawback is that not only the running time but also the space requirement depends exponentially on the parameter value. It seems interesting to find out whether some known methods to "cut down" space requirements in dynamic programming [123, 149, 150, 159] can be used to circumvent this.

Other techniques. Finally, we remark that parameterized algorithmics offers several sophisticated algorithmic tools not explored in voting up to now. Such tools comprise iterative compression [182, 125] or methods based on randomized algorithms such as color-coding [4], divide & color [146], or chromatic coding [3]. For example, it seems reasonable that a divide & color method should also lead to fixed-parameter tractability of Kemeny Score with respect to the number of candidates. Although in this case it seems unlikely that this directly leads to an improved worst-case running time bound, randomized algorithms combined with heuristic speed-up tricks often lead to improvements in practice, as shown in experimental work [48, 136].

12.2.3 Parameter-oriented approaches

We end with an overview of most parameterizations used in this work coming with general "recommendations" and challenging open questions.

Number of votes. Beside some trivial examples for fixed-parameter tractability with respect to the (single) parameter number of votes (such as for Young Score), for voting problems we are only aware of W[1]-hardness or NP-hardness for a constant number of votes. Many of the NP-hardness proofs rely on "encoding" a directed graph into an election (see Section 11.1 for the formal definition). To this end, a range of "tricks" is applied, such as the edge-coloring approach suggested in Section 11.3.2 or dividing edges such that every vertex has either degree two or has only degree-two neighbors [77]. However, a general scheme to encode an arbitrary digraph into an election is not known yet. In particular, it seems interesting whether an arbitrary tournament can be encoded within three votes. Answering this question positively could also be used to show the NP-hardness of Kemeny Score with three votes.

Number of candidates. As shown by various examples in this and other work [82, 99], integer linear programming in combination with Lenstra's result often allows for showing fixed-parameter tractability with respect to the number of candidates. On

the positive side, this offers a nice classification tool. On the negative side, there is no known direct combinatorial algorithm showing fixed-parameter tractability with respect to the number of candidates for non-trivial cases. Herein, by non-trivial we refer to problems for which a solution cannot be obtained by permuting all candidates such as Kemeny Score or manipulation by a single manipulator. The development of combinatorial fixed-parameter algorithms with respect to the number of candidates seems to be of great interest for many voting problems. Moreover, one might hope that having an algorithm for one problem this might be adapted to other problems as well.

Large-value parameters. With this term we refer to parameters for which the values seem to be quite large for many instances. Examples of our work comprise the "Kemeny score" (Chapter 3) or the "total number of undetermined pairs" (Chapter 8) of a partial profile. For such parameters, depth-bounded search tree algorithms seem to provide simple tools to show fixed-parameter tractability. Note that although this does not lead to good bounds of the worst-case running time, there might be reasonable hope for a better performance in practice. To this end, nice features of search trees are that they can be combined with data reduction rules and come with polynomial-time space requirements.

Parameters bounding deficit vectors. The parameterizations that led to the dynamic programming algorithms in Section 6.1 and Section 9.1.2 both provide an upper bound for the sum of the entries of a deficit vector where a deficit vector provides a kind of demand for every candidate. In our examples, the demands correspond to the "number of zero-positions" that a candidate has to assume (for Possible Winner) and the "number of votes in which a candidate has to become worse than the distinguished candidate" (for Dodgson Score). Dealing with a problem that allows for meaningful size-bounded deficit vectors, it might be worth investigating whether the dynamic programming algorithm does apply.

Range parameterizations. For Kemeny Score, we investigated the parameters average and maximum range of a candidate. While the parameterization by the average range led to NP-hardness even for small constant values, we obtained a dynamic programming algorithm showing fixed-parameter tractability with respect to the maximum range. The basic algorithm as stated in the conference version [20] exploits a simple decomposition property similar to the concept of dynamic programming on graphs of bounded "pathwidth":[1] It explores the instance from left to the right and only updates the information at the "interface". This might be an approach of general interest for the parameter "maximum range". Furthermore, it seems desirable to extend this approach by means of kernelization. To this end, a next step would be to investigate whether polynomial kernels are likely to exist.

Average parameterizations. We proposed the average KT-distance as meaningful parameterization for Kemeny Score. Since this parameter captures a natural

[1] For the algorithm coming with an improved running time as stated in Section 3.6 it is not sufficient to have bounded ranges in the input votes but one needs to show that this is also true in the final Kemeny ranking.

property of elections in many situations, it might also be of interest for other voting problems. This seems to be a fruitful field of research, particularly because one can resort to a range of existing techniques: dynamic programming (Section 3.5), partial kernelization (Chapter 4), branching strategies [189], and a further algorithm coming with subexponential running time [142].

Dirtiness parameterizations. In Chapter 4 we introduced the "number of dirty pairs" according to different majorities as a parameterization measuring the amount of agreement based on pairs of candidates. This concept has turned out useful to design partial kernels for several median problems [24]. The identification of further problems allowing for meaningful "dirtiness" measures based on element pairs seems interesting.

Combined parameters. In Chapter 9, we provided an example for combined parameters capturing realistic settings. In the spirit of a multivariate complexity analysis all discussed single parameters can be combined to pairs or tuples of parameters. Although the meaningfulness of a combined parameter depends on the setting and needs to be evaluated carefully, this offers a wide range of studies for future research.

Bibliography

[1] N. Ailon. Aggregation of partial rankings, p-ratings, and top-m lists. *Algorithmica*, 57(2):284–300, 2010. Cited on pp. 40 and 44.

[2] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM*, 55(5), 2008. Article 23 (October 2008). Cited on pp. 24 and 43.

[3] N. Alon, D. Lokshtanov, and S. Saurabh. Fast FAST. In *Proceedings 36th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5555 of *LNCS*, pages 49–58. Springer, 2009. Cited on pp. 18 and 202.

[4] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995. Cited on p. 202.

[5] K. Arrow. *Social Choice and Individual Values*. Cowles Foundation, Yale University Press, 2nd edition, 1963. Cited on p. 13.

[6] K. J. Arrow, A. K. Sen, and K. Suzumura, editors. *Handbook of Social Choice and Welfare*. North-Holland, 2002. Cited on pp. 11 and 14.

[7] Y. Bachrach, N. Betzler, and P. Faliszewski. Probabilistic possible winner determination. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, pages 697–702. AAAI Press, 2010. Cited on pp. vii, 144, and 168.

[8] N. Baigent. Metric rationalisation of social choice functions according to principles of social choice. *Mathematical Social Sciences*, 13(1):59–65, 1987. Cited on p. 83.

[9] J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer, 2nd edition, 2009. Cited on pp. 5 and 181.

[10] J. J. Bartholdi III and J. Orlin. Single transferable vote resists strategic voting. *Social Choice and Welfare*, 8(4):341–354, 1991. Cited on pp. 15, 16, and 18.

[11] J. J. Bartholdi III, C. A. Tovey, and M. A. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6:227–241, 1989. Cited on pp. 15 and 16.

[12] J. J. Bartholdi III, C. A. Tovey, and M. A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989. Cited on pp. 15, 17, 24, 69, 70, 71, 78, 82, and 83.

[13] J. J. Bartholdi III, C. A. Tovey, and M. A. Trick. How hard is it to control an election? *Mathematical and Computer Modelling*, 16(8-9):27–40, 1992. Cited on pp. 4, 15, 17, 173, 174, 175, 177, and 194.

[14] D. Baumeister and J. Rothe. Taking the final step to a full dichotomy of the Possible Winner problem in pure scoring rules. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*, pages 1019–1020 (short paper), 2010. Cited on pp. 3, 87, 88, 91, 122, 123, 163, 165, and 200.

[15] N. Betzler. On problem kernels for possible winner determination under the k-approval protocol. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 6281 of *LNCS*, pages 114–125. Springer, 2010. Cited on pp. ix and 18.

[16] N. Betzler, R. Bredereck, and R. Niedermeier. Partial kernelization for Rank Aggregation: Theory and experiments. In *Proceedings of 5th International Symposium on Parameterized and Exact Computation (IPEC 2010)*, LNCS. Springer, 2010. To appear. Also presented at the *3rd International Workshop on Computational Social Choice (COMSOC)*. Cited on pp. viii and 18.

[17] N. Betzler, R. Bredereck, R. Niedermeier, and J. 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 (SOFSEM)*, 2011. Cited on pp. vii, 198, and 202.

[18] N. Betzler and B. Dorn. Towards a dichotomy of finding possible winners in elections based on scoring rules. In *Proceedings of the 34th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 5734 of *LNCS*, pages 124–136. Springer, 2009. Cited on pp. ix and 18.

[19] N. Betzler and B. 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. Cited on pp. ix, 18, and 163.

[20] N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. Fixed-parameter algorithms for Kemeny scores. In *Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management (AAIM)*, volume 5034 of *LNCS*, pages 60–71. Springer, 2008. Cited on pp. viii, 17, 18, 25, 41, and 203.

[21] N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. Fixed-parameter algorithms for Kemeny rankings. *Theoretical Computer Science*, 410:4554–4570, 2009. Cited on pp. viii, 18, 23, and 25.

[22] N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. How similarity helps to efficiently compute Kemeny rankings. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 657–664, 2009. Cited on pp. viii, 18, 25, 29, 30, 31, 40, and 41.

[23] N. Betzler, M. R. Fellows, C. Komusiewicz, and R. Niedermeier. Parameterized algorithms and hardness results for some graph motif problems. In *Proceedings of the 19th Annual Symposium on Combinatorial Pattern Matching (CPM)*, volume 5029 of *LNCS*, pages 31–43. Springer, 2008. Cited on p. vii.

[24] N. Betzler, J. Guo, C. Komusiewicz, and R. Niedermeier. Average parameterization and partial kernelization for computing medians. In *Proceedings of the 9th Latin American Theoretical Informatics Symposium (LATIN)*, volume 6034 of *LNCS*, pages 60–71. Springer, 2010. Cited on pp. viii, 18, and 204.

[25] N. Betzler, J. Guo, C. Komusiewicz, and R. Niedermeier. Average parameterization and partial kernelization for computing medians. *Journal of Computer and System Sciences*, 2010. doi:10.1016/j.jcss.2010.07.005. Cited on pp. viii, 42, 45, 57, and 58.

[26] N. Betzler, J. Guo, and R. Niedermeier. Parameterized computational complexity of Dodgson and Young elections. In *Proceedings of the 11th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 5124 of *LNCS*, pages 402–413. Springer, 2008. Cited on pp. ix and 17.

[27] N. Betzler, J. Guo, and R. Niedermeier. Parameterized computational complexity of Dodgson and Young elections. *Information and Computation*, 208(2):165–177, 2010. Cited on pp. ix, 17, 78, and 82.

[28] N. Betzler, S. Hemmann, and R. Niedermeier. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 53–58, 2009. Cited on pp. ix, 18, 88, and 94.

[29] N. Betzler and J. Uhlmann. Parameterized complexity of candidate control in elections and related digraph problems. In *Proceedings of the 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA)*, volume 5165 of *LNCS*, pages 43–53. Springer, 2008. Cited on pp. ix, 17, and 198.

[30] N. Betzler and J. Uhlmann. Parameterized complexity of candidate control in elections and related digraph problems. *Theoretical Computer Science*, 410(52):5425–5442, 2009. Cited on pp. ix and 185.

[31] N. Betzler, R. van Bevern, M. R. Fellows, C. Komusiewicz, and R. Niedermeier. Parameterized algorithmics for finding connected motifs in biological networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2010. Cited on p. vii.

[32] T. Biedl, F. J. Brandenburg, and X. Deng. On the complexity of crossings in permutations. *Discrete Mathematics*, 309:1813–1823, 2007. Cited on pp. 12, 24, and 44.

[33] D. Black. On the rationale of group decision-making. *Journal of Political Economy*, 56(1):23–34, 1948. Cited on p. 167.

[34] H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 5917 of *LNCS*, pages 17–37. Springer, 2009. Cited on pp. 8 and 146.

[35] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009. Cited on pp. 8, 146, and 158.

[36] H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. In *Proceedings of the 17th Annual European Symposium on Algorithms (ESA)*, volume 5757 of *LNCS*, pages 635–646. Springer, 2009. Cited on p. 159.

[37] J.-C. Borda. M'emoire sur les 'elections au scrutin. Histoire de l'Academie Royale des Sciences 1781, 1784. Cited on p. 12.

[38] S. Brams and P. Fishburn. Voting procedures. In K. Arrow, A. K. Sen, and K. Suzumura, editors, *Handbook of Social Choice and Welfare*, volume 1, pages 173–236. Elsevier, 2002. Cited on pp. 13 and 88.

[39] S. Brams and R. Sanver. Voting systems that combine voting and preference. In S. Brams, W. Gehrlein, and F. Roberts, editors, *The Mathematics of Preference, Choice, and Order: Essays in Honor of Peter C. Fishburn*, pages 215–237. Springer, 2009. Cited on pp. 13 and 19.

[40] F. Brandt. Some remarks on Dodgson's voting rule. *Mathematical Logic Quarterly*, 55(4):460–463, 2009. Cited on p. 69.

[41] F. Brandt, M. Brill, E. Hemaspaandra, and L. A. Hemaspaandra. Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, pages 715–722, 2010. Cited on pp. 17 and 167.

[42] F. Brandt, M. Brill, and H. G. Seedig. On the fixed-parameter tractability of composition-consistent tournament solutions. In *Proceedings of the 3rd International Workshop on Computational Social Choice (COMSOC)*, pages 43–54, 2010. Cited on p. 18.

[43] F. Brandt, F. Fischer, P. Harrenstein, and M. Mair. A computational analysis of the tournament equilibrium set. *Social Choice and Welfare*, 34(4):597–609, 2010. Cited on p. 13.

[44] R. Bredereck. *Fixed-Parameter Algorithms for Computing Kemeny Scores – Theory and Practice*. Studienarbeit, Universität Jena, 2010. arXiv:1001.4003v1. Cited on p. 31.

[45] R. Bredereck. *Graph and Election Problems Parameterized by Feedback Set Numbers*. Diplomarbeit, Universität Jena, 2010. Cited on pp. 198 and 202.

[46] E. Brelsford, P. Faliszewski, E. Hemaspaandra, I. Schnoor, and H. Schnoor. Approximability of manipulating elections. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 44–49. AAAI Press, 2008. Cited on p. 16.

[47] G. Brightwell and P. Winkler. Counting linear extensions. *Order*, 8(3):225–242, 1991. Cited on p. 144.

[48] S. Bruckner, F. Hüffner, R. M. Karp, R. Shamir, and R. Sharan. Topology-free querying of protein interaction networks. *Journal of Computational Biology*, 17(3):237–252, 2010. Cited on p. 202.

[49] J. Cai, V. T. Chakaravarthy, L. A. Hemaspaandra, and M. Ogihara. Competing provers yield improved Karp-Lipton collapse results. *Information and Computation*, 198(1):1–23, 2005. Cited on p. 159.

[50] I. Caragiannis, J. A. Covey, M. Feldmann, C. M. Homan, C. Kaklamanis, N. Kramikolas, A. D. Procaccia, and J. S. Rosenschein. On the approximability of Dodgson and Young elections. In *Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1058–1067. SIAM, 2009. Cited on pp. 69 and 82.

[51] I. Caragiannis, C. Kaklamanis, N. Karanikolas, and A. D. Procaccia. Socially desirable approximations for Dodgson's voting rule. In *Proceedings of the 11th ACM Conference on Electronic Commerce(EC)*, pages 253–262. ACM, 2010. Cited on p. 69.

[52] J. Chen, Y. Liu, S. Lu, B. O'Sullivan, and I. Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *Journal of the ACM*, 55(5), 2008. Article No. 21. Cited on pp. 18 and 41.

[53] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice (invited paper). In *Proceedings of the 33rd Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 4362 of *LNCS*, pages 51–69. Springer, 2007. Cited on pp. 1, 11, 15, and 16.

[54] Y. Chevaleyre, J. Lang, N. Maudet, and J. Monnot. Possible winners when new candidates are added: the case of scoring rules. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, pages 762–767, 2010. Cited on pp. 15, 88, 166, 167, and 175.

[55] B. Chor, M. Fellows, and D. W. Juedes. Linear kernels in linear time, or how to save k colors in $o(n^2)$ steps. In *Proceedings of the 30th International Workshop on Graph-Theoretice Concepts in Computer Sciences (WG)*, volume 3353 of *LNCS*, pages 257–269. Springer, 2004. Cited on p. 156.

[56] R. Christian, M. R. Fellows, F. A. Rosamond, and A. Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, 2007. Cited on pp. 15, 17, 19, 70, 72, 82, and 200.

[57] V. Conitzer. Computing Slater rankings using similarities among candidates. In *Proceedings of the 19th AAAI Conference on Artificial Intelligence (AAAI)*, pages 613–619. AAAI Press, 2006. Cited on pp. 18, 24, 26, and 58.

[58] V. Conitzer. Making decisions based on the preferences of multiple agents. *Communications of the ACM*, 53(3):84–94, 2010. Cited on pp. 1 and 11.

[59] V. Conitzer, A. Davenport, and J. Kalagnanam. Improved bounds for computing Kemeny rankings. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence (AAAI)*, pages 620–626. AAAI Press, 2006. Cited on pp. 25, 26, 59, 62, and 67.

[60] V. Conitzer, M. Rognlie, and L. Xia. Preference functions that score rankings and maximum likelihood estimation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 109–115, 2009. Cited on p. 24.

[61] V. Conitzer and T. Sandholm. Vote elicitation: Complexity and strategy-proofness. In *Proceedings of the 18th AAAI Conference on Artificial Intelligence (AAAI)*, pages 392–397. AAAI Press, 2002. Cited on p. 14.

[62] V. Conitzer and T. Sandholm. Common voting rules as maximum likelihood estimators. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 145–152. AUAI Press, 2005. Cited on pp. 24 and 26.

[63] V. Conitzer and T. Sandholm. Communication complexity of common voting rules. In *Proceedings of the 6th ACM Conference on Electronic Commerce (EC)*, pages 78–87. ACM, 2005. Cited on pp. 14 and 15.

[64] V. Conitzer and T. Sandholm. Nonexistence of voting rules that are usually hard to manipulate. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence (AAAI)*, pages 627–634, 2006. Cited on p. 198.

[65] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3):1–33, 2007. Cited on pp. 16, 18, 88, 126, 146, 174, 178, and 193.

[66] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, 2001. Cited on pp. 94, 143, 150, and 158.

[67] P. Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. *Theoretical Computer Science*, 351(3):337–350, 2006. Cited on p. 168.

[68] A. Davenport and J. Kalagnanam. A computational study of the Kemeny rule for preference aggregation. In *Proceedings of the 19th AAAI Conference on Artificial Intelligence (AAAI)*, pages 697–702. AAAI Press, 2004. Cited on pp. 25 and 62.

[69] M. J. A. N. de Caritat (Marquis de Condorcet). Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. Paris: L'Imprimerie Royal, 1785. Cited on pp. 5, 12, 24, and 69.

[70] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, 4th edition, 2010. Cited on p. 5.

[71] C. Dodgson. A method of taking votes on more than two issues. Pamphlet printed by the Clarendon Press, Oxford, and headed "not yet published", 1876. Cited on pp. 13 and 69.

[72] M. Dom, J. Guo, F. Hüffner, R. Niedermeier, and A. Truss. Fixed-parameter tractability results for feedback set problems in tournaments. *Journal of Discrete Algorithms*, 8(1):76–86, 2010. Cited on p. 18.

[73] M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and IDs. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5555 of *LNCS*, pages 378–389. Springer, 2009. Cited on p. 159.

[74] B. Dorn and I. Schlotter. Multivariate complexity analysis of swap bribery. In *Proceedings of the 5th International Symposium on Parameterized and Exact Computation (IPEC)*, LNCS. Springer, 2010. To appear. Cited on pp. 19, 146, and 167.

[75] R. G. Downey and M. R. Fellows. Threshold dominating sets and an improved version of W[2]. *Theoretical Computer Science*, 209:123–140, 1998. Cited on p. 187.

[76] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999. Cited on pp. 1, 6, 7, 8, 17, 78, 178, 181, and 195.

[77] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the 10th International World Wide Web Conference (WWW)*, pages 613–622, 2001. Cited on pp. 2, 12, 13, 24, 26, 41, 42, 43, 63, 201, and 202.

[78] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation revisited, 2001. Manuscript. Cited on pp. 24, 29, and 42.

[79] E. Elkind and P. Faliszewski. Approximation algorithms for campaign management. In *Proceedings of the 6th Workshop on Internet and Network Economics (WINE)*, 2010. To appear. Cited on p. 15.

[80] E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. In *Proceedings of the 2nd International Symposium on Algorithmic Game Theory (SAGT)*, volume 5814 of *LNCS*, pages 299–310. Springer, 2009. Cited on pp. 15, 89, and 167.

[81] E. Elkind, P. Faliszewski, and A. Slinko. Cloning in elections. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*, pages 768–773, 2010. Cited on p. 15.

[82] E. Elkind, P. Faliszewski, and A. Slinko. On the role of distances in defining voting rules. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 375–382, 2010. Cited on pp. 18, 83, 201, and 202.

[83] E. Ephrati and J. Rosenschein. The Clarke Tax as a consensus mechanism among automated agents. In *Proceedings of the 9th AAAI Conference on Artificial Intelligence (AAAI)*, pages 173–178, 1991. Cited on p. 11.

[84] E. Ephrati and J. Rosenschein. A heuristic technique for multi-agent planning. *Annals of Mathematics and Artificial Intelligence*, 20(1–4):13–67, 1997. Cited on p. 11.

[85] G. Erdélyi, H. Fernau, J. Goldsmith, N. Mattei, D. Raible, and J. Rothe. The complexity of probabilistic lobbying. In *Proceedings of the 1st International Conference on Algorithmic Decision Theory (ADT)*, volume 5783 of *LNCS*, pages 86–97. Springer, 2009. Cited on p. 19.

[86] G. Erdélyi, L. A. Hemaspaandra, J. Rothe, and H. Spakowski. On approximating optimal weighted lobbying, and frequency of correctness versus average-case polynomial time. In *Proceedings of the 16th International Symposium on Fundamentals of Computation Theory (FCT)*, volume 4639 of *LNCS*, pages 300–311. Springer, 2007. Cited on p. 15.

[87] G. Erdélyi, L. A. Hemaspaandra, J. Rothe, and H. Spakowski. Generalized juntas and NP-hard sets. *Theoretical Computer Science*, 410(38-40):3995–4000, 2009. Cited on p. 16.

[88] G. Erdélyi, M. Nowak, and J. Rothe. Sincere-strategy preference-based approval voting broadly resists control. In *Proceedings of the 33nd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 5162 of *LNCS*, pages 311–322. Springer, 2008. Cited on pp. 17, 173, and 197.

[89] G. Erdélyi and M. R. Fellows. Parameterized control complexity in fallback voting. Technical report, arXiv:1004.3659v1, 2010. Cited on pp. 19, 173, and 197.

[90] G. Erdélyi, L. Piras, and J. Rothe. Control complexity in fallback voting. Technical report, arXiv:1004.3398v1, 2010. Cited on pp. 17, 173, and 197.

[91] G. Erdélyi and J. Rothe. Control complexity in fallback voting. In *Proceedings of Computing: the 16th Australasian Theory Symposium (CATS 2010)*, Australian Computer Society Conferences in Research and Practice in Information Technology Series, pages 39–43, 2010. Cited on pp. 173 and 197.

[92] R. Fagin, R. Kumar, and D. 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. Cited on pp. 2, 12, 24, and 44.

[93] P. Faliszewski. Nonuniform bribery (short paper). In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1569–1572, 2008. Cited on pp. 14, 94, and 167.

[94] P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. How hard is bribery in elections. *Journal of Artificial Intelligence Research*, 35:485–532, 2009. Cited on p. 14.

[95] P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. Multimode control attacks on elections. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 128–133, 2009. Cited on pp. 15, 17, 173, and 197.

[96] P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. Using complexity to protext elections. *Communications of the ACM*, 53(1):74–82, 2010. Cited on pp. 1 and 11.

[97] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Llull and Copeland voting broadly resist bribery and control. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 724–730. AAAI Press, 2007. Cited on pp. 174, 175, 189, and 190.

[98] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Copeland voting fully resists constructive control. In *Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management (AAIM)*, volume 5034, pages 165–176, 2008. Cited on pp. 178, 182, 189, and 200.

[99] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, 35:275–341, 2009. Cited on pp. 9, 14, 15, 17, 19, 94, 125, 167, 173, 174, 175, 177, 198, and 202.

[100] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. A richer understanding of the complexity of election systems. In S. Ravi and S. Shukla, editors, *Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz*, chapter 14, pages 375–406. Springer, 2009. Cited on pp. 1, 11, 15, 167, and 193.

[101] P. Faliszewski, E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. The shield that never was: societies with single-peaked preferences are more open to manipulation and control. In *Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, pages 118–127. ACM, 2009. Cited on p. 17.

[102] P. Faliszewski, E. Hemaspaandra, and H. Schnoor. Copeland voting: Ties matter. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 983–990, 2008. Cited on pp. 19 and 126.

[103] P. Faliszewski and A. Procaccia. AI's war on manipulation: Are we winning? Invited to *AI Magazine* special issue on algorithmic game theory, 2010. forthcoming. Cited on pp. 1, 11, 14, and 16.

[104] M. R. Fellows. Towards fully multivariate algorithmics: some new results and directions in parameter ecology. In *Proceedings of the 20th International Workshop on Combinatorial Algorithms (IWOCA)*, volume 5874 of *LNCS*, pages 2–10. Springer, 2009. Cited on pp. 7 and 68.

[105] M. R. Fellows, D. Hermelin, F. A. Rosamond, and S. Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1):53–61, 2009. Cited on pp. 99 and 100.

[106] M. R. Fellows, B. Jansen, D. Lokshtanov, F. A. Rosamond, and S. 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 (FSTTCS)*, 2010. To appear. Cited on pp. 18, 70, 83, and 146.

[107] M. R. Fellows, D. Lokshtanov, N. Misra, F. A. Rosamond, and S. Saurabh. Graph layout problems parameterized by vertex cover. In *Proceedings of the 19th International Symposium on Algorithms and Computation (ISAAC)*, volume 5369 of *LNCS*, pages 294–305. Springer, 2008. Cited on p. 9.

[108] M. R. Fellows, F. A. Rosamond, and A. Slinko. Sensing God's will is fixed-parameter tractable. Technical Report 561, University of Auckland, 2008. Cited on pp. 70, 83, and 146.

[109] H. Fernau. On parameterized enumeration. In *Proceedings of the 8th International Computing and Combinatorics Conference (COCOON)*, volume 2387 of *LNCS*, pages 564–573. Springer, 2002. Cited on p. 168.

[110] H. Fernau, F. V. Fomin, D. Lokshtanov, M. Mnich, G. Philip, and S. Saurabh. Ranking and drawing in subexponential time. In *Proceedings of the 21st International Workshop on Combinatorial Algorithms (IWOCA)*, LNCS. Springer, 2010. To appear. Cited on p. 18.

[111] J. Fiala, P. A. Golovach, and J. Kratochvíl. Parameterized complexity of coloring problems: Treewidth versus vertex cover. In *Proceedings of the 6th Annual Conference on Theory and Applications of Models of Computation (TAMC)*, volume 5523 of *LNCS*, pages 221–230. Springer, 2009. Cited on p. 9.

[112] J. Flum and M. Grohe. The parameterized complexity of counting problems. *SIAM Journal on Computing*, 33(4):892–922, 2004. Cited on p. 168.

[113] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. Cited on pp. 1, 6, and 7.

[114] L. Fortnow and R. Santhanam. Infeasibility of instance compression and succinct PCPs for NP. In *Proceedings of the 40th ACM Symposium on Theory of Computing (STOC)*, pages 133–142. ACM, 2008. Long version to appear in *Journal of Computer and System Sciences*. Cited on pp. 8 and 158.

[115] A. Frank and É. Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7:49–65, 1987. Cited on p. 9.

[116] E. Friedgut, G. Kalai, and N. Nisan. Elections can be manipulated often. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 243–249. IEEE Computer Society, 2008. Cited on pp. 16 and 198.

[117] W. Gaertner. *A Primer in Social Choice Theory*. LSE Perspectives in Economic Analysis. Oxford University Press, revised edition, 2009. Cited on pp. 11 and 13.

[118] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979. Cited on pp. 5, 91, 132, and 178.

[119] J. Geanakoplos. Three brief proofs of Arrow's impossibility theorem. *Economic Theory*, 26(1):211–215, 2005. Cited on p. 13.

[120] M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Elicitation strategies for soft constraint problems with missing preferences: Properties, algorithms and experimental studies. *Artificial Intellingence*, 174(3-4):270–294, 2010. Cited on p. 14.

[121] A. Gibbard. Manipulation of voting schemes: A general result. *Econometrica*, 41(4):587–601, 1973. Cited on pp. 14 and 16.

[122] J. Gramm, R. Niedermeier, and P. Rossmanith. Fixed-parameter algorithms for Closest String and related problems. *Algorithmica*, 37(1):25–42, 2003. Cited on p. 9.

[123] S. Guillemot and F. Sikora. Finding and counting vertex-colored subtrees. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 6281 of *LNCS*, pages 405–416. Springer, 2010. Cited on p. 202.

[124] J. Guo, F. Hüffner, and R. Niedermeier. A structural view on parameterizing problems: distance from triviality. In *Proceedings of the 1st International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 3162 of *LNCS*, pages 162–173. Springer, 2004. Cited on p. 127.

[125] J. Guo, H. Moser, and R. 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. Cited on p. 202.

[126] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007. Cited on pp. 8 and 146.

[127] G. Hägele and F. Pukelsheim. The electoral writings of Ramon Llull. *Studia Lulliana*, 41(97):3–18, 2001. Cited on p. 12.

[128] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Hybrid elections broaden complexity-theoretic resistance to control. *Mathematical Logic Quarterly*, 55(4):397–424, 2009. Cited on p. 173.

[129] E. Hemaspaandra and L. A. Hemaspaandra. Dichotomy for voting systems. *Journal of Computer and System Sciences*, 73(1):73–83, 2007. Cited on pp. 16, 18, 88, 89, 123, 126, and 167.

[130] E. Hemaspaandra, L. A. Hemaspaandra, and J. 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. Cited on pp. 6, 15, 16, 69, 77, and 82.

[131] E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, 171(5-6):255–285, 2007. Cited on pp. 15, 17, 146, 173, 174, 175, 177, 194, and 195.

[132] E. Hemaspaandra, H. Spakowski, and J. Vogel. The complexity of Kemeny elections. *Theoretical Computer Science*, 349(3):382–391, 2005. Cited on pp. 6, 16, 24, and 40.

[133] S. Hemmann. *Komplexität der Bestimmung Alleiniger Möglicher Gewinner bei Borda- und Maximin-Verfahren*. Diplomarbeit, Universität Jena, 2008. Cited on p. 136.

[134] C. M. Homan and L. A. Hemaspaandra. Guarantees for the success frequency of an algorithm for finding Dodgson-election winners. *Journal of Heuristics*, 15(4):403–423, 2009. Cited on p. 69.

[135] F. Hüffner, N. Betzler, and R. Niedermeier. Separator-based data reduction for signed graph balancing. *Journal of Combinatorial Optimization*, 20(4):335–360, 2010. Cited on p. vii.

[136] F. Hüffner, S. Wernicke, and T. Zichner. Algorithm engineering for color-coding with applications to signaling pathway detection. *Algorithmica*, 52(2):114–132, 2008. Cited on p. 202.

[137] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. Cited on p. 7.

[138] B. N. Jackson, P. S. Schnable, and S. Aluru. Consensus genetic maps as median orders from inconsistent sources. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(2):161–171, 2008. Cited on pp. 2, 12, 13, and 24.

[139] D. Jungnickel. *Graphs, Networks and Algorithms*. Springer, 3rd edition, 2008. Cited on pp. 5 and 190.

[140] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12:415–440, 1987. Cited on p. 9.

[141] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the 12th ACM Symposium on Theory of Computing (STOC)*, pages 302–309. ACM Press, 1980. Cited on p. 159.

[142] M. Karpinski and W. 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 (ISAAC)*, 2010. To appear. Cited on pp. 18, 23, 25, 26, 29, 31, 32, 42, 43, 45, and 204.

[143] J. Kemeny. Mathematics without numbers. *Daedalus*, 88:571–591, 1959. Cited on p. 23.

[144] C. Kenyon-Mathieu and W. Schudy. How to rank with few errors. In *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC)*, pages 95–103. ACM, 2007. Cited on p. 25.

[145] J. Kleinberg and É. Tardos. *Algorithm Design*. Addison Wesley, 2006. Cited on pp. 24 and 37.

[146] J. Kneis, D. Mölle, S. Richter, and P. Rossmanith. Divide-and-color. In *Proceedings of the 32nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 4271 of *LNCS*, pages 58–67. Springer, 2006. Cited on p. 202.

[147] C. Komusiewicz, R. Niedermeier, and J. Uhlmann. Deconstructing intractability—a multivariate complexity analysis of Interval Constrained Coloring. *Journal of Discrete Algorithms*, 2010. To appear. Cited on p. 200.

[148] K. Konczak and J. Lang. Voting procedures with incomplete preferences. In *Proceedings of the IJCAI-2005 Multidisciplinary Workshop on Advances in Preference Handling*, 2005. Cited on pp. 14, 16, and 88.

[149] I. Koutis. Faster algebraic algorithms for path and packing problems. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5125 of *LNCS*, pages 575–586. Springer, 2008. Cited on p. 202.

[150] I. Koutis and R. Williams. Limits and applications of group algebras for parameterized problems. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5555 of *LNCS*, pages 653–664. Springer, 2009. Cited on p. 202.

[151] J. Köbler and O. Watanabe. New collapse consequences of NP having small circuits. *SIAM Journal on Computing*, 28(1):311–324, 1998. Cited on p. 159.

[152] J. Lang. Vote and aggregation in combinatorial domains with structured preferences. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1366–1371, 2007. Cited on p. 15.

[153] J. Lang, M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Winner determination in sequential majority voting. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1372–1377, 2007. Cited on p. 88.

[154] H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983. Cited on pp. 8, 9, and 17.

[155] A. Levenglick. Fair and reasonable election systems. *Behavioral Science*, 20(1):34–46, 1975. Cited on p. 23.

[156] C. Lindner and J. Rothe. Fixed-parameter tractability and parameterized complexity applied to problems from computational social choice. Supplement in the *Mathematical Programming Glossary*, A. Holder, editor. INFORMS Computing Society, October 2008. Cited on p. 17.

[157] H. Liu, H. Feng, D. Zhu, and J. Luan. Parameterized computational complexity of control problems in voting. *Theoretical Computer Science*, 410(27-29):2746–2753, 2009. Cited on pp. 17, 19, and 198.

[158] H. Liu and D. Zhu. Parameterized complexity of control problems in maximin election. *Information Processing Letters*, 110(10):383–388, 2010. Cited on pp. 19 and 173.

[159] D. Lokshtanov and J. Nederlof. Saving space by algebraization. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 321–330. ACM, 2010. Cited on p. 202.

[160] M. Mahajan and V. Raman. Parameterizing about guaranteed values: MaxSat and MaxCut. *Journal of Algorithms*, 31(2):335–354, 1999. Cited on p. 43.

[161] M. Mahajan, V. Raman, and S. Sikdar. Parameterizing above or below guaranteed values. *Journal of Computer and System Sciences*, 75:137–153, 2009. Cited on pp. 23, 25, 42, and 43.

[162] D. Marx. Closest substring problems with small distances. *SIAM Journal on Computing*, 38(4):1382–1410, 2008. Cited on p. 42.

[163] J. C. McCabe-Dansted. Approximability and computational feasibility of Dodgson's rule. Master's thesis, University of Auckland, 2006. Cited on pp. 17, 70, 72, and 83.

[164] J. C. McCabe-Dansted, G. Pritchard, and A. Slinko. Approximability of Dodgson's rule. *Social Choice and Welfare*, 31(2):311–330, 2008. Cited on p. 69.

[165] C. McCartin. Parameterized counting problems. *Annals of Pure and Applied Logic*, 138(1-3):147–182, 2006. Cited on p. 168.

[166] D. McGarvey. A theorem on the construction of voting paradoxes. *Econometrica*, 21(4):311–330, 1953. Cited on p. 176.

[167] I. McLean. The Borda and Condorcet pronciples: three medieval applications. *Social Choice and Welfare*, 7:99–108, 1989. Cited on p. 12.

[168] I. McLean and A. Urken. *Classics of Social Choice*. University of Michigan Press, Ann Arbor, Michigan, 1995. Cited on p. 11.

[169] R. Meir, A. D. Procaccia, J. S. Rosenschein, and A. Zohar. The complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research*, 33:149–178, 2008. Cited on p. 17.

[170] T. Meskanen and H. Nurmi. Closeness counts in social choice. In M. Braham and F. Steffen, editors, *Power, Freedom, and Voting*, pages 289–306. Springer, 2008. Cited on p. 83.

[171] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006. Cited on pp. 1, 6, 8, 68, 127, and 156.

[172] R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS'10)*, volume 5 of *LIPIcs*, pages 17–32. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2010. Cited on pp. 7, 68, 127, and 200.

[173] R. Niedermeier and P. Rossmanith. A general method to speed up fixed-parameter-tractable algorithms. *Information Processing Letters*, 73:125–129, 2000. Cited on p. 68.

[174] N. Nishimura, P. Ragde, and D. M. Thilikos. Parameterized counting algorithms for general graph covering problems. In *Proceedings of the 9th Workshop on Algorithms and Data Structures (WADS)*, volume 3608 of *LNCS*, pages 99–109. Springer, 2005. Cited on p. 168.

[175] H. Nurmi. *Comparing Voting Systems*. Kluwer Academic Publishers, 1987. Cited on p. 13.

[176] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994. Cited on pp. 5 and 123.

[177] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Incompleteness and incomparability in preference aggregation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1464–1469, 2007. Cited on p. 88.

[178] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Incompleteness and incomparability in preference aggregation: Complexity results. *Artificial Intelligence – Special Issue on Representing, Processing, and Learning Preferences: Theoretical and Practical Challenges*, 2009. Accepted subject to minor revision. Cited on pp. 168 and 201.

[179] A. Procaccia and J. S. Rosenschein. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research*, 28:157–181, 2007. Cited on pp. 16 and 198.

[180] G. Pruesse and F. Ruskey. Generating linear extensions fast. *SIAM Journal on Computing*, 23(2):373–386, 1994. Cited on p. 144.

[181] V. Raman and S. Saurabh. Improved fixed parameter tractable algorithms for two "edge" problems: MAXCUT and MAXDAG. *Information Processing Letters*, 104(2):65–72, 2007. Cited on p. 29.

[182] B. A. Reed, K. Smith, and A. Vetta. Finding odd cycle transversals. *Operations Research Letters*, 32(4):299–301, 2004. Cited on p. 202.

[183] J. Rothe, H. Spakowski, and J. Vogel. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems*, 36(4):375–386, 2003. Cited on pp. 6, 16, 69, 71, and 82.

[184] M. A. Satterthwaite. Strategy-proofness and Arrow's conditions. *Journal of Economic Theory*, 10:187–217, 1975. Cited on pp. 14 and 16.

[185] F. Schalekamp and A. van Zuylen. Rank aggregation: Together we're strong. In *Proceedings of the 11th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 38–51. SIAM, 2009. Cited on pp. 25, 59, 62, and 63.

[186] N. Schwarz. *Rank aggregation by Criteria—Minimizing the maximum Kendall-tau distance*. Diplomarbeit, Universität Jena, 2009. Cited on p. 44.

[187] D. Sculley. Rank aggregation for similar items. In *Proceedings of the 7th SIAM International Conference on Data Mining (SDM)*, pages 587–592, 2007. Cited on pp. 2, 13, and 24.

[188] M. Shindler. Approximation algorithms for the Metric k-Median problem. Written Qualifying Exam Paper, University of California, Los Angeles. Cited on p. 44.

[189] N. Simjour. Improved parameterized algorithms for the Kemeny aggregation problem. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 5917 of *LNCS*, pages 312–323. Springer, 2009. Cited on pp. 18, 23, 25, 26, 29, 31, 32, 42, 43, 44, 62, 67, and 204.

[190] A. Taylor. *Social Choice and the Mathematics of Manipulation*. Cambridge University Press, 2005. Cited on pp. 11 and 13.

[191] M. Truchon. An extension of the Condorcet criterion and Kemeny orders. Technical report, cahier 98-15 du Centre de Recherche en Économie et Finance Appliquées, Université Laval, Québec, Candada, 1998. Cited on pp. 30 and 52.

[192] A. van Zuylen and D. P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. *Mathematics of Operations Research*, 34:594–620, 2009. Cited on pp. 24 and 68.

[193] T. Walsh. Uncertainty in preference elicitation and aggregation. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 3–8. AAAI Press, 2007. Cited on pp. 88, 125, and 167.

[194] L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 196–201. AAAI Press, 2008. Cited on pp. 18, 88, 89, 92, 95, 96, 127, 137, 146, 165, 166, and 167.

[195] L. Xia and V. Conitzer. Generalized scoring rules and the frequency of coalitional manipulability. In *Proceedings of the 9th ACM Conference on Electronic Commerce (EC)*, pages 109–118, 2008. Cited on p. 16.

[196] L. Xia and V. Conitzer. A sufficient condition for voting rules to be frequently manipulable. In *Proceedings of the 9th ACM Conference on Electronic Commerce (EC)*, pages 99–108, 2008. Cited on p. 16.

[197] L. Xia and V. Conitzer. Determining possible and necessary winners under common voting rules given partial orders. Unpublished manuscript, 2010. Cited on p. 137.

[198] L. Xia, V. Conitzer, and J. Lang. Voting on multiattribute domains with cyclic preferential dependencies. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 202–207, 2008. Cited on p. 15.

[199] L. Xia, V. Conitzer, and A. D. Procaccia. A scheduling approach to coalitional manipulation. In *Proceedings of the 11th ACM Conference on Electronic Commerce (EC)*, pages 275–284. ACM, 2010. Cited on pp. 16, 88, and 126.

[200] L. Xia, M. Zuckerman, A. D. Procaccia, V. Conitzer, and J. S. Rosenschein. Complexity of unweighted coalitional manipulation under some common voting rules. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 348–353, 2009. Cited on pp. 16, 19, and 126.

[201] H. Young and A. Levenglick. A consistent extension of Condorcet's election principle. *SIAM Journal on Applied Mathematics*, 35(2):285–300, 1978. Cited on pp. 2, 13, and 24.

[202] H. P. Young. Social choice scoring functions. *SIAM Journal on Applied Mathematics*, 28(4):824–838, 1975. Cited on p. 13.

[203] H. P. Young. Extending Condorcet's rule. *Journal of Economic Theory*, 16:335–353, 1977. Cited on pp. 13, 17, 69, and 70.

[204] M. Zuckerman, A. D. Procaccia, and J. S. Rosenschein. Algorithms for the coalitional manipulation problem. *Artificial Intelligence*, 173(2):392–412, 2009. Cited on pp. 16, 126, and 198.

Ehrenwörtliche Erklärung

Hiermit erkläre ich,

- dass mir die Promotionsordnung der Fakultät bekannt ist,
- dass ich die Dissertation selbst angefertigt habe, keine Textabschnitte oder Ergebnisse eines Dritten oder eigene Prüfungsarbeiten ohne Kennzeichnung übernommen und alle von mir benutzten Hilfsmittel, persönliche Mitteilungen und Quellen meiner Arbeit angegeben habe,
- dass ich die Hilfe eines Promotionsberaters nicht in Anspruch genommen habe und dass Dritte weder unmittelbar noch mittelbar geldwerte Leistungen von mir für Arbeiten erhalten haben, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen,
- dass ich die Dissertation noch nicht als Prüfungsarbeit für eine staatliche oder andere wissenschaftliche Prüfung eingereicht habe,
- dass ich weder die gleiche, eine in wesentlichen Teilen ähnliche, noch eine andere Abhandlung bereits bei einer anderen Hochschule als Dissertation eingereicht habe.

Jena, July 2010 *Nadja Betzler*