Complexity of Strategic Influences in Elections and Group Identification with a Main Focus on Incomplete Information

DISSERTATION

zur Erlangung des Grades eines Doktors

rer. pol.

der Fakultät III - Wirtschaftswissenschaften, Wirtschaftsinformatik
und Wirtschaftsrecht der Universität Siegen

vorgelegt von

Dipl.-Math. oec. Christian Reger

Erstgutachter: Prof. Dr. Gábor Erdélyi
Zweitgutachter: Prof. Dr. Ulf Lorenz

Datum der Disputation: 19. Dezember 2018

Dekan der Fakultät III: Prof. Dr. Volker Wulf

Abstract

Traditionally, an election consists of a candidate set and a voter set, and each voter completely ranks all candidates from first to last. Depending on the voters' rankings, a voting rule determines the winner or the winners of the election. Apart from winner determination, there are further problems related to voting which are usually formulated as decision problems. For instance, in *bribery* we ask whether an external agent can make a candidate a winner of the given election by changing at most a certain number of voters' votes.

In this thesis, we generalize the assumption of complete information and regard nine different partial information models in total (some of them are already known). We study the complexity of the problems *necessary winner* [117], *possible winner* [117], *necessary bribery*, and *possible bribery* in the voting rules k-Approval and k-Veto.

Moreover, we consider another model of incomplete information. In contrast to the models previously mentioned, lottery-based voting is based on the assumption that the votes are given as complete rankings, but a lottery draws at random a voter subset of fixed and predetermined size to which a voting rule is applied afterward. Once more, we investigate the complexity for variants of necessary and possible winner as well as necessary and possible bribery. Besides, we consider a counting problem asking how many subelections of a certain size exist such that a designated candidate wins the election.

We further regard two voting rules frequently used in practice—Plurality with Runoff and Veto with Runoff. We explore the complexity of bribery and several control problems such as *control by adding, deleting, and replacing candidates and/or voters*. For several problems, we assume that there is full information.

Last but not least, we attend to group identification. On the one hand, we determine the complexity for group bribery as well as three different destructive group control variants in group identification. On the other hand, we consider partial information in group identification and, in particular, we study the problems *possibly qualified individuals* and *necessarily qualified individuals* as well as variations of these problems where each individual must qualify exactly k individuals.

Zusammenfassung

Gewöhnlich besteht eine Wahl aus einer Kandidatenmenge sowie einer Wählermenge, wobei jeder Wähler alle Kandidaten vom ersten bis zum letzten Rang einreiht. Abhängig von den Rankings der Wähler bestimmt eine Wahlregel den oder die Gewinner der Wahl. Neben der Gewinnerbestimmung gibt es weitere Probleme bezüglich Wahlen, welche traditionell als Entscheidungsprobleme formuliert werden. Im *Bestechungsproblem* stellt sich beispielsweise die Frage, ob ein externer Agent einen Kandidaten zum Gewinner der Wahl machen kann, wenn höchstens eine gewisse Anzahl an Wählerstimmen geändert wird.

In dieser Arbeit verallgemeinern wir die Annahme vollständiger Information und betrachten insgesamt neun verschiedene Modelle partieller Information (manche dieser Modelle sind schon bekannt). Wir untersuchen die Komplexität für die Probleme *Necessary Winner*, [117], *Possible Winner* [117], *Necessary Bribery* und *Possible Bribery* für die Wahlregeln k-Approval und k-Veto.

Wir betrachten außerdem ein weiteres Modell unvollständiger Information. Im Gegensatz zu den vorher genannten Modellen basiert lotteriebasiertes Wählen auf der Annahme, dass die Wählerstimmen zwar als vollständige Rankings gegeben sind, aber ein Lotterie eine Wählerteilmenge fester vorgegebener Größe zufällig auslost, auf welche anschließend eine Wahlregel angewandt wird. Erneut untersuchen wir unter anderem die Komplexität für Varianten der Probleme Necessary und Possible Winner sowie Necessary und Possible Bribery. Überdies betrachten wir ein Zählproblem, bei dem sich die Frage stellt, auf wie viele verschiedene Möglichkeiten die Lotterie eine feste Anzahl an Wählern auslosen kann, so dass ein designierter Kandidat ein Wahlgewinner ist.

Ferner betrachten wir zwei häufig in der Praxis verwendete Wahlregeln, Pluralität und Veto jeweils mit Stichwahl. Wir erforschen die Komplexität von Bestechung sowie mehrerer Kontrollprobleme wie *Kontrolle durch Hinzufügen, Löschen und Ersetzen von Kandidaten und/oder Wählern*. Für sämtliche Probleme nehmen wir an, dass volle Information vorliegt.

Schließlich widmen wir uns der Gruppenidentifikation. Wir bestimmen einerseits die Komplexität für Gruppenbestechung sowie drei verschiedene Arten destruktiver Gruppenkontrolle. Andererseits betrachten wir Gruppenidentifikation unter partieller Information und untersuchen insbesondere die Probleme *Possibly Qualified Individuals* und *Necessarily Qualified Individuals* sowie Varianten dieser Probleme, bei denen jedes Individuum genau k Individuen qualifizieren muss.

Acknowledgements

First of all, I have to thank my supervisor Gábor Erdélyi for giving me the chance to write my phd thesis, for introducing me into COMSOC and into doing research, for giving me valuable feedback, and for various jointly published papers. Moreover, I am grateful to Ulf Lorenz who agreed to be the second referee of my thesis.

I further would like to thank my co-authors Yongjie Yang, Andreas Pfandler, and Dirk Briskorn.

In this context, I have to thank Stephan Meisel and my proofreaders Stefan and Tomáš for giving me valuable advises and feedback.

Of course, I am grateful to the University of Siegen and the DFG (ER 738/2 -1) for funding and supporting my research.

Finally, I have to thank Jenni and my family for encouraging me, for their patience while I was writing this thesis, and for standing by my side.

List of Publications

Parts of this thesis have already been published. This book is especially based on the publications listed below. The results from these publications listed from 1. to 4. under "Conference Publications" can be found in Chapter 8, 7, 3, and 3 in this thesis, respectively. The respective chapters are presented very similarly to the respective papers (so this thesis is somewhere located between monograph and cumulative form); nevertheless, these chapters may contain additional results, additional proofs (omitted in the respective papers), or had to be adjusted in any other thinkable form, in order to adapt it to the common framework in this book. Aside from these publications, the thesis contains some additional results, partially published in workshop proceedings or mentioned in talks.

Conference Publications

1. **Complexity of Group Identification With Partial Information**. G. Erdélyi, C. Reger, and Y. Yang. Proceedings of the *5th International Conference on Algorithmic Decision Theory (ADT 2017)*, Luxembourg. Springer-Verlag *Lecture Notes in Artificial Intelligence*, pages 182-196, October 2017.

2. **The Complexity of Bribery and Control in Group Identification**. G. Erdélyi, C. Reger, and Y. Yang. Proceedings of the *16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*, Sao Paulo, Brazil. International Foundation of Autonomous Agents and Multiagent Systems (IFAAMAS), pages 1142-1150, May 2017.

3. **Possible Bribery in k-Approval and k-Veto Under Partial Information**. G. Erdélyi and C. Reger. Proceedings of the *17th International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA 2016)*, Varna, Bulgaria. Springer-Verlag *Lecture Notes in Artificial Intelligence 9883*, pages 299-309, September 2016.

4. **Bribery in k-Approval and k-Veto Under Partial Information (Extended Abstract)**. D. Briskorn, G. Erdélyi, and C. Reger. Proceedings of the *15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016)*, Singapore. International Foundation of Autonomous Agents and Multiagent Systems (IFAAMAS), pages 1299-1300, May 2016.

Workshop Publications

Concerning workshop publications, we present some results of the M-PREF publication. The second and third paper are not subject in this thesis, albeit mentioned few times. Both publications ap-

peared in the scope of our research about strategic influences in elections under partial information. Nevertheless, we omit them and focus on necessary and possible winner and bribery instead. The fourth publication is a preliminary version of our paper at AAMAS 2016 ("Bribery in k-Approval and k-Veto Under Partial Information").

1. **Completing the Puzzle: Solving Open Problems for Control in Elections**. G. Erdélyi, C. Reger, and Y. Yang. Proceedings of the *11th Multidisciplinary Workshop on Advances in Preference Handling (M-PREF18)*, workshop notes, New Orleans, USA, February 2018.

2. **Possible Voter Control in k-Approval and k-Veto Under Partial Information**. G. Erdélyi, and C. Reger. Proceedings of the *1st Workshop Präferenzen und Personalisierung in der Informatik (PPI 2017)*, workshop notes, Stuttgart, Germany. GI-Edition, Lecture Notes in Informatics, pages 185-192, March 2017.

3. **Voter Control in k-Approval and k-Veto Under Partial Information**. C. Reger. Proceedings of the *14th International on Artificial Intelligence and Mathematics (ISAIM 2016)*, workshop notes, Fort Lauderdale, FL, USA, January 2016.

4. **Bribery Under Partial Information**. D. Briskorn, G. Erdélyi, and C. Reger. Proceedings of the *2nd Workshop on Exploring Beyond the Worst Case in Computational Social Choice (EXPLORE 2015)*, workshop notes, Istanbul, Turkey, May 2015.

Talks

At each conference and workshop listed so far (except for the second conference publication), I presented our results by giving a talk (or a poster presentation, at AAMAS'16). Our results about bribery and control in group identification were presented by Yongjie Yang at AAMAS'17 in Sao Paolo.

This thesis is based on further talks at the workshops QBWL'14 (Meinerzhagen/Germany, title: *Computational Social Choice*) and CASC'14 (Bad Belzig/Germany, title: *Bribery Under Partial Information*). In the former talk, I gave a short introduction to COMSOC, talked about the models Gaps, FP, TOS, and PC, and provided some results of necessary bribery for these four models. In the latter talk, I additionally presented results about the structures CEV and 1TOS and provided the complexity of necessary bribery under the voting rules k-Approval ($k \neq 2$) and k-Veto ($k \neq 3$) and some further results in possible bribery.

Last but not least, I presented various results of this thesis at the Doctoral Consortium in Luxemburg (at the conference ADT'17). At this talk, I highlighted, amongst others, our results about necessary and possible bribery as well as about necessary and possible winner.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Voting applies to many situations in everyday life whensoever individual preferences are aggregated in order to reach a collective decision. This embraces decisions such as which restaurant to go to, which film to watch, or which holiday destination to choose. All these examples share the common goal that a ranking over the possible alternatives should be determined or at least a consensus should be reached. The actual group decision depends on the underlying voting rule or social choice function. Consider further a setting where a group tries to determine which group members are qualified for a given task and which ones not. The actual decision frequently depends on the way the group members evaluate each other. Voting also turns out to be a useful tool in the design of recommender systems [93], planning [62], mechanism design [61], ranking algorithms [51], collaborative filtering [137], similarity search [75], location planning [10], or machine learning [166], just to name a few. Of course, voting applies to political elections as well, e.g., when a president or a coalition of different governing parties is elected, or when voters vote for a bill.

Voting theory itself, but also similar streams of research—such as fair division [153], group identification [110], or judgment aggregation [17]—are unified under the more and more upcoming area of research called *Computational Social Choice* or *COMSOC*, for short. COMSOC is an interdisciplinary and heterogeneous field somewhere located between political sciences, economics, computer science, mathematics, game theory, and philosophy. This list makes no claim to be complete.

In computer science applications, we deal with huge data volumes. Therefore, it makes sense to study the computational properties and especially the computational complexity of various problems related to voting.

Without going further into detail, an election (C,V) consists of a set C of candidates, a set V of voters, and each voter canonically ranks the candidates from first to last. Given an election, a voting rule determines the winner or the winners, depending on the context. One of the most basic problems in voting theory is the *Winner* problem [15] which asks whether or not a distinguished candidate is a winner of a given election. In practice, computing the winners of an election should be a computationally easy task, even for large elections. This actually seems to be the case for most voting rules used in practice. Unfortunately, there exist some different ways to tamper with the outcome of an election. One such problem is *Bribery* [79] where an external agent—the *briber*—alters some voters' votes in order to make a certain candidate win the election or to prevent a despised candidate from winning. Another way to influence an election is given by *Electoral Control* [16]. This

diction covers many different problems. By way of example, in *Control by Adding Voters* an external agent—the *chair*—adds some new voters to a given election in order to reach his goal. In *Control by Deleting Voters*, the chair deletes some voters from the election. One can define *Control by Adding Candidates* and *Control by Deleting Candidates* analogously to control by adding/deleting voters in a sense that the chair adds and deletes candidates (instead of voters), respectively. Control and bribery have many real-world applications. So control by adding or deleting voters applies to decreasing or increasing the voting age, voter encouragement, organizing get-out-the-vote drives, voter suppression, or withdrawing the voting right from some voters or voter groups. By contrast, candidate control models scenarios such as introducing spoiler candidates, candidate suppression, or representing new candidates. Bribery is self-explanatory and displays scenarios where a malicious agent pays money to some voters in order to change their votes. Variations of bribery and some related problems also apply to lobbying or campaign management [25, 35, 77, 150]. The standard bribery model (with unit prices) is further tailored to situations where an external agent tries to find out how many voters have to be bribed or convinced to change their votes. A third class of strategic behavior in elections is given by *Manipulation* [14, 38] which is not subject of this thesis. Loosely speaking, in manipulation one or more voters—the *manipulators*—coordinate their votes and submit possibly dishonest ballots in order to make a given candidate win or to prevent that candidate from winning an election. As an example, consider two-party systems in political elections where there are two big parties and several small parties without any chance of winning. Although many voters actually favor a small party, they give their votes to one of the two big parties.

Such strategic attacks on elections are undesirable and consequently should be impossible or at least improbable. Unfortunately, control or bribery are possible for many voting rules (cf. [146, 28]). Moreover, the famous theorem by Gibbard and Satterthwaite [94, 149] states that manipulation is possible for almost all common voting rules when there are at least three candidates. The seminal papers by Bartholdi et al. [13, 14, 16] suggest that NP-completeness of such problems, formulated as decision problems, provides a (worst-case) barrier against manipulative attacks. For instance, in case it is computationally hard to decide whether or not bribery of a given voting rule is possible at all, the voting rule offers a certain protection against bribery. Conversely, if the bribery problem is computationally easy, the voting rule is far from being bribery-proof. Many problems for frequently used voting rules have been shown to be easy so far [146, 28]. However, this holds for the canonical assumption that all voters provide complete rankings over all candidates. This in turn does not appear to be realistic in practice. Intuitively, a briber or a chair faces a more difficult problem when the votes are incomplete.

Accordingly, we focus on partial information models in voting. In particular, we study bribery under partial information and examine which impact partial information has on the complexity of bribery, compared to the standard model of full information. We merely consider the constructive variants of bribery asking whether the briber can make a distinguished candidate win the election. Note that the destructive versions are similarly defined, but the briber's goal is to prevent a given candidate from winning. Our analysis is further restricted to two families of voting rules, namely k-Approval and k-Veto ($k \in \mathbb{N}$). These voting rules belong to the family of *scoring rules* defined in a way that voters assign points to candidates according to a fixed scheme—a *scoring vector*—and the candidates with the highest score are the winners (there may be one or more winning candidates). In k-Approval, each voter assigns one point to his k most favorite candidates and zero points to the remaining ones. Similarly, in k-Veto each voter gives zero points to his k least favorite candidates,

and one point to the remaining ones. These voting rules possibly belong to the most important and prominent voting rules and have many applications in practice. Especially the rule 1-Approval (aka Plurality) is a basic rule often used when some people try to reach a decision, based on their favorite alternatives, and the alternative with the most first places wins. Moreover, both classes are closed families and enable us to achieve dichotomy results, that is, we can partition $\mathbb{N}$ into two subsets N_1 and N_2 containing all k with easy and hard bribery problem, respectively. We do not study manipulation for several reasons. Firstly, bribery and control belong to external influences on elections, whereas manipulation corresponds to internal influences. Note that a briber and a chair actually influence and modify an election, whereas in manipulation some voters just misreport their preferences and vote strategically, but submit feasible ballots. Hence, in manipulation there is no illegal or illegitimate behavior in a strong sense although a good voting rule should impede or even preclude reporting dishonest preferences. Last but not least, recently after us, Dey et al. [46] studied manipulation under a setting of partial information where each vote is given as a partial order over the candidates.

This *partial orders* model—we refer to this model as *Pairwise Comparisons* or *PC*, for short, in this thesis—is one of the nine models studied by us. Intuitively, each voter specifies his preferences over some pairs of distinct candidates, for other pairs of different candidates it is unknown whether a voter prefers the one or the other. PC can be regarded as the standard partial information model in voting so far. Konczak and Lang [117] were the first who allowed votes to be partial orders instead of complete rankings. They defined the *possible/necessary winner* problems which ask whether the incomplete votes in an election can be extended to complete rankings such that a distinguished candidate wins for at least one (possible winner) or for all (necessary winner) extensions, respectively. According to [20, 21, 167], necessary winner under partial orders is computationally easy for all scoring rules, whereas possible winner is only easy for 1-Veto (aka Veto), 1-Approval (aka Plurality), some equivalent scoring rules reachable by positive-affine transformations (cf. Chapter 3 for more details), and constant scoring rules (where each candidate is trivially a winner receiving the same score from each voter). For all other non-constant scoring rules, possible winner under partial orders is hard.

In this thesis, we study necessary bribery and possible bribery, defined as combinations of standard bribery with necessary/possible winner. These problems hence generalize bribery under full information as well as necessary/possible winner. As pointed out, we study these problems (1) in a constructive setting, (2) for nine different partial information models, and (3) for the voting rules k-Approval and k-Veto by either showing that a problem formulated as a decision problem is NP-hard (that is, computationally hard in the worst-case) or belongs to P (that is, the problem can be decided in polynomial time).

Different questions encouraged us to perform an extensive complexity study. Firstly, we check whether partial information increases the complexity of some/all/most problems. For example, since possible winner is only easy for Plurality and Veto under partial orders, we know that possible bribery under partial orders can be easy at all merely for these two rules and is computationally hard for k-Approval and k-Veto, $k \geq 2$. Accordingly, we check whether possible bribery becomes hard for Plurality and Veto, too. Secondly, partial orders has been the standard model up to now, but this structure seems to be very general and possibly too general in some contexts. Therefore, Conitzer et al. [39] proposed a general partial information model allowing any structure of partial information. In their setting, votes are given as partial profiles and the *information set* of a partial

profile contains all complete voter profiles that do not contradict the given partial profile. Opposed to the partial orders model, their model in turn is hold as general as possible. In contrast to Conitzer et al. [39], we regard the partial orders model as well as other and more special ways to display partial information, revisiting some other structures that have been proposed or studied in literature meanwhile. For instance, voters may rank only their favorite candidates, only their least favorite candidates, or both. Such votes are known under the name *top-truncated orders*, *bottom-truncated orders*, and *doubly-truncated orders*, respectively [18]. Likewise, one may consider an election with complete information (or with almost complete information), and suddenly new voters or candidates join the election about whom nothing is known. Such structures have been proposed in [117]. Besides, Chevaleyre et al. [33] studied the possible winner problem for the case where new candidates join a given election. Moreover, voters may be indifferent between some candidates or some alternatives may be just incomparable to each other. All in all, we define three new models and regard and formalize six other models already proposed or studied in literature. We provide a complete picture over these nine structures, point out their interrelationships (some models are special cases of other models, other models are incomparable to each other), and examine whether the complexity for a given problem differs between the models, that is, there are models for which a problem is easy, and for the remaining models, the same problem is hard. Another intriguing question is whether the two families k-Approval and k-Veto offer the same or similar protection against bribery under partial information, that is, whether for both voting rule families about the same amount of additional hardness arises when the votes are partial. We further provide the complexities for the possible and necessary winner problem for k-Approval and k-Veto under all models, except for two cases which are still open. Studying problems related to possible/necessary winner appeared to us as a natural first step towards examining strategic behavior in voting under uncertainty the more so as we follow the largest part of literature as well as these two approaches can be interpreted as optimistic and pessimistic approaches. Nevertheless, considering more refined ways to deal with partial information seems to be promising. So one could study probabilistic approaches (see e.g., [7]), perform quantitative analyses (under given statistical data, is a candidate rather a winner or rather not?), or define other, more fine-grained criteria to estimate the winner(s) of an incomplete election (one such approach is preference elicitation, cf. [159]).

There is yet another way to model incompleteness in voting. In *lot-based voting* [160], the votes are given as complete rankings, but there is a lottery that arbitrarily draws a fixed number of votes. To this random selection of votes, a voting rule is applied to determine the winner of the subelection. Such randomized voting rules have already had a long tradition. By way of example, the doge in Venice was elected by such a voting protocol. Accordingly, there also exists the name *Venetian Elections*. In huge political elections as well as in small polls within institutions, it may occur that the number of voters showing up at the election is more or less known, but not which voters. Secondly, lot-based voting applies to elections where a fixed number of representatives are drawn at random who submit their votes. For example, consider a commission of fixed size that shall vote upon a given proposal, but the commission's members are not known in advance. Lot-based voting further applies to elections where only a small sample is actually accounted for. We again study variations of possible/necessary winner for k-Approval and k-Veto asking whether a distinguished candidate wins for at least one/all subelections consisting of K voters. Besides, we consider the problem *Evaluation* (proposed in [160] for lot-based voting rules and studied for some other voting rules). In evaluation, we ask whether or not a candidate wins with a probability

greater than a certain threshold. Additionally, we investigate a counting version of possible winner (how many subelections do exist such that a candidate c wins?), necessary bribery, and possible bribery (can the briber bribe some voters such that c wins with probability one/greater than zero?). Once again, we provide a complexity study and check whether the additional step—drawing some voters at random—increases the computational complexity compared to the standard setting and thus impedes strategic behavior.

Where we introduce strategic behavior in elections under incomplete information on the one hand, there have yet been various open problems under full information for some voting rules on the other hand. Such unsolved questions have existed in bribery, *Replacement Control* [17, 123], *Multimode Control by Adding/Deleting Voters* [81], and *Multimode Control by Adding/Deleting Candidates* [81] for Plurality with Runoff and Veto with Runoff (for formal definitions, we refer to Chapter 2). Once again, our focus lies on the constructive variants of these problems. In multimode control by adding/deleting voters, the chair adds some new voters and deletes some previous voters. Likewise, multimode control by adding/deleting candidates is defined. In replacement control, the chair either replaces some voters with other voters (*Control by Replacing Voters*) or some candidates with other candidates (*Control by Replacing Candidates*). These two problems are defined in a similar manner to multimode control, but in replacement control the number of deleted voters/candidates must be equal to the number of added voters/candidates.

Basically, both voting rules are defined analougsly to their counterparts without a runoff stage—Plurality and Veto. In a first step, the candidates' scores are determined. After this, the two best candidates compete against each other in the runoff (possibly, there may be ties and a tie-breaking rule must decide which candidates move to runoff). The runoff stage offers a certain additional quality check. Note that in Plurality or Veto, a candidate less preferred to each other candidate by a majority of voters may still be the unique winner. Such a candidate is also called the *Condorcet loser*. E.g., consider a Plurality election where two voters rank candidate c first and 20 other voters rank c last, while each of twenty other candidates is ranked first by exactly one of these voters. Verify that c is the unique Plurality winner with two points, but 20 of 22 voters regard c as the worst possible outcome of the election. The Plurality with Runoff rule never selects c as a winner, as c reaches the runoff, but loses against the other candidate in the runoff, regardless of which other candidate follows c to the runoff.

Variants of the voting rule Plurality with Runoff are often used in political elections (such as the presidential election in France), sometimes in multistage versions. Veto with Runoff and slightly modified variants of this rule are known from competitions of any kind (e.g. sports competitions) over several rounds where in each round one or more candidates are ruled out. Or consider a situation where a group tries to reach a decision and the group members agree to restrict themselves to the two best alternatives in a sense that they make as few voters as possible totally dissatisfied, that is, the two alternatives worth considering have the fewest vetoes among all alternatives.

We consider yet another setting related to voting, called *group identification*. Recall the example with a situation where a group of individuals try to determine which group members are qualified for a given task and which ones not. In a sense, group identification is a model related to voting where the sets of voters and candidates coincide (and are called *individuals*) and none of them provides a ranking over the other individuals, but has a binary opinion about all individuals including himself. Besides, the model of group identification premises that each individual is honest and has no incentive to misrepresent his opinion about all individuals (nevertheless, such settings would be

interesting for future research). Formally, N is a set of n individuals, and for each two individuals a and a' it holds that either *a qualifies* or *disqualifies* a'. A *profile* φ is a mapping describing for each pair (a,a') of individuals whether a qualifies or disqualifies a'. Last but not least, a *social rule* determines a subset of N which we refer to as *socially qualified individuals*. Depending on the evaluations of the individuals in N, the number of socially qualified individuals may be arbitrary.

The model of group identification has been studied in economics [47, 48, 134, 147]. The social rules considered in this thesis are *consent rules* and the two *procedural rules* denoted by *consensus-start-respecting rule* and *liberal-start-respecting rule*. These rules have been extensively investigated in the literature [47, 130, 147]. The question whether or not an individual is socially qualified depends on his self-evaluation as well as on the evaluation of other individuals. For some social rules, a negative self-evaluation of an individual may even lead to his social disqualification. Likewise, a possible self-evaluation may directly imply or at least facilitate the social qualification of an individual.

Group identification applies to many different contexts. It is suitable in situations where a group of individuals shall decide who in this group has a certain identity and who not [111]. Moreover, the model applies to scenarios where a group of people try to find out who in this group is suitable for a given task. Assume for instance that in a university department a group of faculty members elect who in this group should become the dean for the next period of time. Or consider a set of automated agents or robots that must collaborate to perform a given task. For some economical reasons, merely few agents should execute the task. In such cases, a joint decision must be reached of which agents should perform the job. Note that such settings where automated agents collaborate become more and more important in times of Industry 4.0.

Finally, group identification can reflect group-dynamic behavior or group structures. In this context, campaign management, lobbying, or strategic behavior in general appear to be worth studying. While group identification had been studied concerning some axiomatic properties or from the sociological point out view, Yang and Dimitrov [171] initiated the study on the complexity of strategic influences in form of constructive control by adding, deleting, and partitioning individuals. This inspired us to continue this emerging research and study destructive control as well as constructive and destructive bribery.

Furthermore, we deviate from the canonical setting with complete information. Previous works on group identification premise that an external agent knows for each pair (a,a') of individuals whether a qualifies or disqualifies a'. Or, equivalently speaking, a *complete profile* is given. In practice, this does not appear to be realistic. Suppose for example that an agent or a company wishes to forecast the outcome of a social rule, but not all the information is accessible, which seems a reasonable assumption especially when the group is large. Therefore, we study the effect of *partial profiles* and investigate the complexity of the two problems *Necessarily Qualified Individuals* and *Possibly Qualified Individuals* (NQI and PQI, for short). Intuitively, a partial profile over the individuals in N is defined in a way that an individual a qualifies an individual a', disqualifies a', or it is unknown whether a qualifies or disqualifies a'. Analogously to partial preferences in elections, a profile ϕ extends a partial profile φ when ϕ is a complete profile not contradicting the partial profile. The NQI (PQI) problem asks whether or not a subset S of individuals is qualified under social rule f for every complete profile ϕ (at least one profile ϕ) extending the partial profile φ. We examine the complexity for both problems and various social rules under two assumptions. Under the first assumption, each individual may qualify an arbitrary number of individuals, whereas under

the second assumption, every individual qualifies exactly k individuals (with $k \in \mathbb{N}$).

Organization

This thesis is organized as follows. While Part I deals with voting with a focus on partial information, Part II is about group identification.

In Chapter 2, we provide some preliminaries in mathematics, complexity theory, and graph theory. Finally, we give a short introduction to voting theory.

Chapter 3 is about voting under partial information. In a first step, we define nine partial information models. We further point out the interconnections of these structures. In the remainder of this chapter, we investigate the complexity of the problems possible/necessary winner and possible/necessary bribery.

The subsequent chapter—Chapter 4—presents yet another way to study elections under incomplete information. Analogously to the standard setting, all voters completely rank the candidates from first to last, but a lottery selects at random a voter subset of fixed size. We study possible and necessary winner, a counting problem where we count the number of winning subelections of a given candidate, evaluation, and two bribery variants.

Chapter 5 is about voter control, candidate control, and bribery in the rules Plurality with Runoff and Veto with Runoff. In contrast to the previous two chapters, we merely study several problems under complete information.

As pointed out, Part II, is about group identification. Chapter 6 gives an introduction to group identification, whereas in Chapter 7 we study strategic behavior in group identification for various social rules. Chapter 8 deals with group identification under incomplete information.

Last but not least, the Appendix provides some omitted calculations, proofs, and examples.

Part I

Voting

Chapter 2

Preliminaries

In this chapter, in Section 2.1, 2.2, and 2.3, we provide some basics in complexity theory, graph theory, and voting theory, respectively.

First of all, we recall some basics in mathematics. Although we assume the reader to be familiar with the most important notions in mathematics, we subsequently present some of them, especially those for which there exist different dictions in literature. By way of example, to avoid confusion, we let $\mathbb{N} = \{1,2,3,\ldots\}$ denote the set of *natural numbers*. Besides, $\mathbb{N}_0 := \mathbb{N} \cup \{0\} = \{0,1,2,\ldots\}$ contains all *nonnegative integers*. We further assume the reader to be familiar with the sets $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$. For two rational numbers a and b, we let $[a,b]$ denote the *rational interval* throughout this thesis, that is $[a,b] := \{x \in \mathbb{Q} : \ a \leq x \leq b\}$. Analogously, we define $(a,b] := \{x \in \mathbb{Q} : \ a < x \leq b\}$. Likewise, we obtain $[a,b)$ and (a,b). Note that intervals can be empty, e.g., when $a > b$ holds. We restrict ourselves to rational numbers as irrational or transcendent numbers only play a minor role in informatics. Rational numbers/intervals suffice for our purposes since numbers in $\mathbb{R} \setminus \mathbb{Q}$ can merely be approximated by computer programs.

We premise that the most basic set-theoretic notions are known to the reader. To avoid confusion, we denote the case where A is a *subset* of B by $A \subseteq B$ or $A \subset B$. Although many computer scientists write $A \subset B$ meaning that $A \subseteq B$ and $A \neq B$, we adapt the traditional notation known from mathematics where both notions are identical. We write $A \subsetneq B$ which is equivalent to $A \subseteq B$ and $A \neq B$. In this case, we say that A is a *proper subset* of B. Finally, $|A| := |\{a : \ a \in A\}|$ denotes the number of elements in A or, equivalently, the *cardinality* or *size* of A. In literature, one can also find expressions such as $||A||$, $\#A$, or $\mathrm{card}(A)$. The *power set* $\mathscr{P}(A) := \{B : \ B \subseteq A\}$ of A is defined as the set containing all subsets of A. Note that if A is finite, $\mathscr{P}(A)$ contains $2^{|A|}$ elements, including A itself and $\emptyset$ (the empty set). Finally, $A \dot{\cup} B$ denotes the *disjoint set union* of two sets A and B, implicitly assuming that $A \cap B = \emptyset$ holds. Notice that this notion can be easily extended to more than two sets.

We briefly describe some sets of integers as follows. For two integers k and l, the set $\{k,\ldots,l\}$ is equivalent to the set $\{j \in \mathbb{Z} : k \leq j \leq l\}$. E.g., $\{4,\ldots,9\}$ is an abbreviated representation of $\{4,5,6,7,8,9\}$. We sometimes write expressions like $4 \leq j \leq 9$ containing all integers from 4 to 9. For the sake of convenience, we further let $[l] := \{1,\ldots,l\}$ for a natural number l.

Last but not least, we assume the reader to familiar with the *Landau symbol* $O(f)$ for a given function f. We say that a function "g is in $O(f)$" or "$g = O(f)$" when g asymptotically grows no faster than f. Throughout this thesis, we often use expressions such as "a problem is in $O(n^2)$"

meaning that the running time of an algorithm solving or deciding the problem is bounded from above by a polynomial quadratic in the problem's input size (cf. Section 2.1).

2.1 Complexity Theory

This section provides some basic complexity theoretic notions and is widely based on the books [89, 145]. Generally, studying the complexity of a problem aims at determining how much amount of time or space is required in the worst-case until a solution of the problem is found or until we find out that a solution does not exist. Basically, we distinguish between algorithms having a polynomial-time running time in the worst-case and between algorithms that require exponentially many steps until a solution is computed, detected, or until we find out that a solution does not exist. In both cases, polynomial or exponential time refers to the input size of the studied problem. For example, when the input of a problem has size $n \in \mathbb{N}$ and the algorithm to solve the problem has worst-case running time 2^n, the number of steps a computer program has to perform exponentially grows in n. For small instances, this provides no difficulty for current computers, but huge input data might considerably increase even a modern computer's running time, compared to algorithms that run in polynomial time.

In the following, we present a formal model that describes problem instances and solutions in terms of languages over a fixed and finite *alphabet* Σ. Typically, we have $\Sigma = \{0,1\}$. Let us start by defining some fundamental notions.

Definition 2.1. *Let Σ be a finite alphabet.*

(a) By Σ^n, we denote the set of all finite strings *or* words *of length $n \in \mathbb{N}_0$ over alphabet Σ.*

(b) $\Sigma^* := \bigcup_{n\in\mathbb{N}_0} \Sigma^n$ contains all strings over Σ with arbitrary length.

(c) A language *L is defined as a subset of Σ^*.*

Let us specify now what is meant by a problem [89].

Definition 2.2. *A* problem Π *consists of a list of input parameters and a statement or question describing the properties and shape a* solution *or* answer *to the problem must have.*

The statement can be given as a statement in the classical meaning as well as a question of which the answer is either YES or NO. Assume for example that our input consists of two rational numbers x and y, and the statement is "Compute $x+y$". Another example might have $x,y \in \mathbb{Q}$ as its input and the statement/question might be "Maximize xy under the constraint $0 \leq x,y \leq 1$". Besides, the input could be identical to the one of the previous example, but contains a nonnegative integer B, in addition. The statement might then be a question such as "Do there exist any x,y with $0 \leq x,y \leq 1$ such that $xy \geq B$?"

Most problems in this thesis are defined in the style of this latter example and are called *decision problems*. In order to formally define decision problems, we first introduce the notion *instance*.

Definition 2.3. *An instance of a problem is a concrete and feasible assignment of values for the given input parameters [89].*

E.g., when a problem's input is "A set M with $M \subseteq \{1,\ldots,10\}$", possible instances are $\{1,2,5\}$, $\{10\}$, $\{1,\ldots,10\}$, or $\emptyset$.

Decision problems [89] are defined as follows.

Definition 2.4. *Let D_Π denote the set of all possible instances of a decision problem Π and let $I \in D_\Pi$ be any instance. Then an algorithm deciding Π either outputs $I \in Y_\Pi \subseteq D_\Pi$ ("I is a YES-instance") or $I \in N_\Pi := D_\Pi \setminus Y_\Pi$ ("I is a NO-instance").*

There is a connection between decision problems and languages over an alphabet Σ in a way that each YES-instance—and in particular the input parameters of any given instance—can be expressed by a finite string over a given language L. Strings in Σ^* either (1) belong to Y_Π (the YES-instances), (2) belong to N_Π (the NO-instances), or (3) do not correspond to instances of Π (one might refer to them as the infeasible instances) and consequently, the latter kind of instances formally belong to $\Sigma^* \setminus D_\Pi$ [89].

Next, we specify what we mean by an algorithm [89].

Definition 2.5. *An* algorithm *is a well-defined procedure that step by step solves (or tries to solve) a given problem.*

There are many different and yet similar definitions of algorithms in literature. One can also consider an algorithm as a function mapping a given instance (a language $D_\Pi \subseteq \Sigma^*$) to a solution or to YES/NO (when the underlying problem is a decision problem; we generally suppose that each instance yields the answer YES or NO although one can define algorithms that do not come to a halt).

As already mentioned, complexity theory aims at classifying a problem Π in terms of its running time. Intuitively, problem instances with larger input size tend to require more running time than instances with smaller input size. We define the input size of a problem as follows.

Definition 2.6.

(a) The input size *or* input length *of an instance I of a problem Π is defined as the number of symbols encoding I by some particular* encoding scheme.

(b) The time complexity function *of an algorithm deciding a problem Π maps a nonnegative integer n to a natural number n', where n' is the largest possible running time or, equivalently speaking, the highest possible number of computational steps that our algorithm requires over all instances encoded by a string of length n.*

The *time complexity* of an algorithm intuitively provides a worst-case measure measuring the number of computational steps (sometimes also called *elementary operations*) of an algorithm deciding our problem until we obtain an answer to our problem. There exist other notions such as *space complexity*, but these are widely irrelevant for our purposes the more so as a polynomial running time implies that the amount of space required to find a solution for a problem is polynomially restricted as well (although the back direction does not hold in general) [89].

The following definition literally follows from [145] and formalizes the concept of a complexity class.

Definition 2.7. *A* complexity class *is defined as a set of problems that can be solved by means of algorithms according to a computational model. These algorithms require no more than a certain amount of the respective complexity resource.*

As pointed out, the "complexity resource" relevant for our purposes is running time. The two basic complexity classes considered are given by P and NP.

Definition 2.8. *The class* P *is defined as the set of problems that can be decided in polynomial time by a* deterministic Turing machine *(*DTM*, for short).*

A Turing machine is an abstract model which simulates an algorithm or a computer program and is—loosely speaking—defined via an input tape (containing the string encoding the input of a problem), a set of states, and some working tape(s) [89]. The latter two simulate the processing of an algorithm solving problem Π on the given input string, initially placed on the input tape. DTMs can model any algorithm or computer program. Or, in other words, any algorithm can be modeled by an equivalent Turing machine [89].

Definition 2.9. *The class* NP *is defined as the set of problems which can be computed in polynomial time by a* non-deterministic Turing machine *(*NDTM*, for short).*

For formal definitions of Turing machines and some examples, we refer to [89, 145]. Recall that an algorithm is *deterministic* if in each stage of the algorithm, the next computational step is uniquely defined. Otherwise, an algorithm is said to be *non-deterministic*. A NDTM is a generalization of a DTM in a sense that it additionally possesses a guessing stage. Or, informally, a NDTM can verify in polynomial time whether or not a guessed candidate for a solution actually corresponds to a solution.

The intuitions behind these two concepts are as follows. Where programs modeled in terms of a DTM require polynomially many steps and the "computation path" (that is, the sequence of computational steps) is uniquely determined, there may be different computation paths for NDTM. Or, equivalently speaking, for each configuration in a NDTM, there may exist different successive configurations.

In general, Turing machines either halt in an accepting state, in a rejecting state, or never halt. We denote Turing machines that always halt on an accepting or rejecting state with *acceptors* as they accept or reject depending on the input. We will observe that all algorithms in this thesis, tailored to decision problems, halt.

Aside from deciding problems, Turing machines can also be used to compute functions, such as the sum of two integers or functions outputting all solutions of a given problem.

Definition 2.10. *The class of functions that can be calculated in polynomial time by a DTM algorithm is called* FP *[157].*

The class P contains problems that can be efficiently decided, whereas NP includes polynomial-time decidable problems as well as intractable problems with exponential running time or problems for which no deterministic polynomial-time algorithm has been discovered yet. Most of such problems are intuitively regarded as *intractable* or *hard*. Formally, it holds $\mathrm{P} \subseteq \mathrm{NP}$. The question "$\mathrm{P} = \mathrm{NP}$ or $\mathrm{P} \neq \mathrm{NP}$" is still open. Although many researchers believe that $\mathrm{P} \neq \mathrm{NP}$ holds, a formal proof is still missing. Observe that under the assumption $\mathrm{P} \neq \mathrm{NP}$, each DTM can be written as a NDTM as well, the back direction does not hold.

All these concepts are defined in an encoding-independent manner. We may do so although there may be different possibilities how to *encode* the input a given problem (we say *encoding scheme*), either over the same alphabet or over different alphabets. By way of example, in case our input is a natural number, possible encodings of the number 13 are 13 (in the decimal system), *D* (in the hexadecimal system), 15 (in the octal system), and 1000 (in the binary system). Observe that the input size is varying between the different encoding schemes. At first sight, it hence appears a little astonishing that we may argue independently from any particular encoding scheme since the complexity of a problem is defined dependent on the input size which in turn depends on the particular encoding scheme used. Coming back to our example, the input—the natural number 13—is larger in the binary system than in the decimal or hexadecimal system and likewise the running times for the decimal or hexadecimal system are smaller than under the binary system. The language $L(\Pi, e)$ of a problem—that is, the YES-instances Y_Π encoded by e—actually refers to a special encoding scheme e [89]. We may yet overlook the particular encoding scheme as all "reasonable" encoding schemes differ from each other in polynomial time. More precisely, the length of any input x encoded by encoding scheme e differs from the lengths of all other encodings of this input in polynomial time. Although Garey and Johnson [89] stated that there is no formal definition of a "reasonable encoding", inputs traditionally (1) should be *consise* (i.e., not padded with unnecessary information), (2) input numbers should be encoded in binary or by any larger fixed base, but not in unary, and (3) any algorithm solving a problem Π should be able to extract the information encoded by a particular encoding scheme in polynomial time [89]. These three requirements seem rational and not really limiting. In case the first condition is hurt, it may be possible that the input itself artificially has exponential size and thus problems requiring exponentially many steps wrongfully become polynomial-time solvable. The same holds for many problems when encoded in unary. Verify that all inputs in our problems studied are given by a "reasonable" encoding scheme. We therefore will not mention this anymore and provide all complexity results in an encoding-independent manner. By way of example, when our input is an election (C,V) together with a designated candidate, we describe this input via listing m candidates in candidate set C, n voters in voter set V ($m \in \mathbb{N}$, $n \in \mathbb{N}_0$), $c \in C$, and the voters' complete rankings over C, by default. Hence, the input size has around $O(m) + O(n) + O(1) + O(mn) = O(mn)$ symbols (ignoring commas, dots, or parentheses helping us to uniquely define our input). We point out that there is a stream of research assuming that a voting problem's input is succinctly represented (cf. [86]). E.g., to evaluate the Plurality rule, it suffices that each voter and the candidate ranked first by this voter are given in the input. In contrast, the canonical setting requires each voter to provide a complete ranking over all candidates.

As a next step, we provide a tool for showing that one problem is at least as hard as another problem. Compare [145] for the following definitions.

Definition 2.11. *Let $\mathscr{C}$ be a complexity class, Π_1 and Π_2 be two problems (or, in terms of languages, two subsets of Σ^* over the same alphabet Σ). We say that* Π_1 many-one reduces to Π_2 *(formally $\Pi_1 \leq_m^p \Pi_2$) if and only if there exists a function $f \in$ FP (that is, the computation of the function can be modeled by a DTM program) such that the condition [$x \in \Pi_1 \iff f(x) \in \Pi_2$] holds for each $x \in \Sigma^*$. Informally, Π_2 is at least as hard as Π_1. We further say that we reduce Π_2 from Π_1.*

The following definition specifies *hardness* of a given problem.

Definition 2.12. *Let* Π_1 *and* Π_2 *be two problems.*

(a) Π_1 *is said to be* $\mathscr{C}$-hard *if* $\Pi_2 \leq_m^p \Pi_1$ *holds for each* $\Pi_2 \in \mathscr{C}$.

(b) Π *is called* $\mathscr{C}$-complete *if and only if* Π *is* $\mathscr{C}$*-hard and* $\Pi \in \mathscr{C}$.

Given $\mathscr{C} = \text{NP}$, we obtain the definitions of NP-hardness and NP-completeness. Definition 2.12 will turn out useful for the following reason. Suppose that a problem Π_1 is NP-complete and $\Pi_1 \leq_m^p \Pi_2$ holds. Then we know that Π_2 is NP-hard.

One problem known to be NP-complete is EXACT COVER BY 3-SETS (X3C, for short) [89].

EXACT COVER BY 3-SETS	
Given:	A set $B = \{b_1, \ldots, b_{3m}\}$ and a collection $\mathscr{S} = \{S_1, \ldots, S_n\}$ with $S_i \subseteq B$, $\lvert S_i \rvert = 3$ for every $i \in [n]$.
Question:	Does $\mathscr{S}$ contain an exact cover for B, or, equivalently, is there a subcollection $\mathscr{S}' \subseteq \mathscr{S}$ such that each element of B occurs in exactly one member of $\mathscr{S}'$)?

We point out that X3C is still NP-complete when each element in B occurs in exactly three sets in $\mathscr{S}$ [97]. We refer to this problem as RESTRICTED EXACT COVER BY 3-SETS (RX3C, for short). Note that it holds $n = 3m$, that is, $\mathscr{S}$ and B both have $3m$ elements.

Another problem known to be NP-complete is THREE-DIMENSIONAL MATCHING (3DM, for short) [89].

THREE-DIMENSIONAL MATCHING	
Given:	Three disjoint sets $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_n\}$, and $Z = \{z_1, \ldots, z_n\}$, and a set of triples $E \subseteq X \times Y \times Z$.
Question:	Does E contain a subset E' such that each element in $X \cup Y \cup Z$ belongs to exactly one triple in E'?

We call E' a three-dimensional matching. This problem generalizes the famous marriage problem [102] for which two genders are given, some of them find each other acceptable, and we try to compute an assignment of women and men finding each other acceptable such that each man (woman) is assigned to exactly one woman (man). Hence, we can interpret a three-dimensional matching as a situation with three genders (X, Y, and Z) and each triple in E corresponds to a combination of one representative of each gender accepting each other.

3DM remains NP-complete under the restriction that each element $a \in X \cup Y \cup Z$ occurs in at most three triples in E [89]. We denote this instance by (≤ 3)-3DM. Observe that this instance is still hard when each element a belongs to either two or three triples in E. We may argue like this for the following reason. If some element $a \in X \cup Y \cup Z$ does not occur in any triple in E, we immediately discard our instance. In case a belongs to exactly one triple, this triple necessarily belongs to a three-dimensional matching (if existing). Step by step, we may therefore fix all triples including such elements until we either arrive at a matching and accept, or reject (as some element does not belong to any triple anymore), or until all elements belong to either two or three triples. A more detailed description of this simplification rule can be found in the Appendix.

This instance has already been used in [32, 33]. We denote the variant of 3DM, where each element is contained in two or three triples in E, by RESTRICTED THREE-DIMENSIONAL MATCHING (R3DM, for short).

Another problem known to be NP-complete is the HAMILTONIAN PATH problem [89] defined in Section 2.2.

While problems in P are easy in a sense that there is further insight into the structure of the problem [89], NP-complete problems can only be verified by guess- and check algorithms in polynomial time (that is, we can check in polynomial time whether a guessed solution yields the answer YES or NO), but an exhaustive search or complete enumeration is necessary in the worst-case. We point out that a problem is also hard when even the solution itself can merely be described in exponential time depending on the input size. One such problem is listing all subsets of a set with n elements ($n \in \mathbb{N}$). Our model of P/NP-hardness does not capture this. One way to deal with this is given by *counting hardness* defined in the following.

In case a problem is NP-complete, counting the number of solutions is intuitively hard as well (as it is even hard to "count" whether or not the number of solutions in greater than zero). Conversely, whenever counting the number of solutions of the a decision problem is easy, the decision problem itself is easy as well. Valiant [157] introduced a formal concept in order to define the *counting complexity* of a given problem:

Definition 2.13. *A* counting Turing machine *(*CTM*, for short) is a NDTM with an auxiliary output device which prints on an additional tape the* number *of YES-instances according to the input string.*

Analogously to DTM and NDTM, the time complexity function of this CTM maps a nonnegative integer n to the largest possible running time over all inputs of size n. Loosely speaking, one can imagine a program that exhaustively guesses all feasible inputs, checks whether the answer to the given problem is YES or NO, and increases the counter on an extra tape counting the number of solutions by one. Similarly to the class NP, Valiant defined the class #*P* tailored to counting problems [157]:

Definition 2.14. *The class #P (pronounced "sharp P") contains all functions for which there is a CTM with a polynomial time complexity function.*

The class # P includes all counting problems solvable by calculating the number of accepting computation paths of a NDTM.

Analogously to decision problems, Valiant defined a hardness notion specific to counting problems. To describe this concept, we first have to present another class of Turing machines, so-called *oracle Turing machines* (*OTMs*, for short). An OTM is a NDTM that consists of a *query tape*, an *answer tape*, and some additional *working tapes*. The NDTM sends some queries to the oracle by writing a string on the query tape, goes to an oracle-state, and then outputs the answer in a single step. This answer is printed on the answer tape. The Turing machine moves to an answer state then. One can consider an oracle as a black box that answers in one step a query sent by the NDTM.

Hardness for counting problems is defined as follows.

Definition 2.15. *A problem* Π *is* # P*-hard if and only if* $\#P \subseteq \mathrm{FP}^{\Pi}$*, where* FP^{Π} *is defined as the class of functions that can be calculated by OTMs from* FP *with oracles for* Π*.*

Similarly to the definition of completeness specific to decision problems, we can define completeness for counting problems:

Definition 2.16. *A counting problem* Π *is* # P*-complete if and only if* $\Pi \in$ # P *and* Π *is* # P*-hard.*

When reducing from hard-to-count problems, we make use of *parsimonious reductions*, i.e., polynomial time reductions that preserve the number of solutions (for details, cf. [89]). Notice that parsimonious reductions generalize many-one reductions for decision problems, that is, they hold for decision problems as well.

One problem known to be hard to count is #PERFECT MATCHINGS FOR BIPARTITE GRAPHS defined in Section 2.2.

Throughout this thesis, we only show hardness. Membership in NP or # P immediately follows as in each hardness proof, a simple guess & check algorithm verifies whether or not a guessed solution is a solution to our problem. We do not state this explicitly in our proofs.

Whenever we indicate the worst-case complexity of a problem by means of the Landau symbol, e.g., we state that a given problem is in $O(m^2n)$, this only provides an upper bound on the running time of a given algorithm. Although many of our algorithms yield instances actually requiring this running time, there may exist proofs for which $O(m^2n)$ is a rough upper bound on the complexity of the given problem and can possibly be improved. Nevertheless, since we merely want to show membership in P, we abstain from providing the best possible worst-case running time for a given algorithm.

When showing membership in P for a given problem related to voting or group identification, we follow the largest part of the literature in COMSOC and provide polynomial-time algorithms in a scarce and economic manner. Accordingly, all numbers in our proofs can be represented by a polynomial-time binary presentation; we frequently renounce of determining the exact complexity, such as $O(mn)$. In Part II, we go a little more into detail as we often distinguish whether or not some given parameters are constant. We further do so in order to stay consistent with the underlying papers.

2.2 Graph Theory

In this section, we introduce some concepts known from graph theory. For the interested reader, we refer to the books [11, 101, 162]. First of all, let us start with some basic notions:

Definition 2.17. *A* graph *is defined as a pair* $G = (\mathcal{V}, \mathcal{E})$*, where* $\mathcal{V}$ *and* $\mathcal{E}$ *are the* vertex set *and* edge set*, respectively. Both* $\mathcal{V}$ *and* $\mathcal{E}$ *are finite.*

- *If* $\mathcal{E} \subseteq (\mathcal{V} \times \mathcal{V}) \setminus \{(x,x) : x \in \mathcal{V}\}$*, the graph is called* directed*. We say* digraph *or* directed graph*.*
- *If* $\mathcal{E} \subseteq \{\mathcal{V}' \subseteq \mathcal{V} : |\mathcal{V}'| = 2\}$*, the graph is said to be* undirected*.*

Given a digraph, we denote edges by (x,y) and the direction matters. We say that the edge starts in x and ends in y or the edge goes from x to y. In particular, the edge (x,y) is different from the edge (y,x) in digraphs. In contrast, in undirected graphs we denote edges by $\{x,y\}$ and the edges $\{x,y\}$ and $\{y,x\}$ are identical. Note that our definition does not allow edges $\{x,x\}$ or (x,x). Such edges are called *loops*. We will assume that graphs do not contain loops.

We can generalize the setting in Definition 2.17 by allowing $\mathcal{E}$ to be a multiset. In other words, each edge (x,y) or $\{x,y\}$ may appear more than once in $\mathcal{E}$.

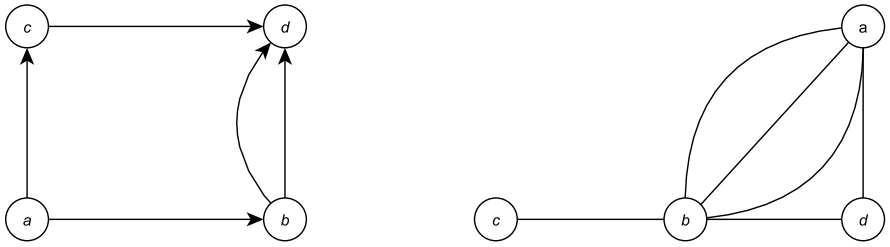

Figure 2.1: The left graph is a digraph, whereas the right graph is undirected. Both graphs contain multiple edges and therefore are not simple.

Definition 2.18. *Let $G = (\mathscr{V}, \mathscr{E})$ be a directed or undirected graph.*

(a) A multiple edge *e is an edge that occurs at least twice and finitely many times in $\mathscr{E}$.*

(b) A multigraph *G is a graph for which multiple edges are allowed.*

(c) G is said to be simple *when it contains no multiple edges.* [1]

We will often make use of the dictions "graph" and "multigraph" interchangeably as many definitions hold for multigraphs and simple graphs. In order to emphasize that a graph must not be a multigraph, we use the diction "simple graph".

Notice that each simple graph is a multigraph as well, but not the other way around, in general. Graphs according to Definition 2.17 are simple. Figure 2.1 presents examples of a digraph and an undirected graph.

In fact, there are two ways to define multigraphs. Firstly, one can define multigraphs as graphs according to Definition 2.17 and each e is additionally assigned a nonnegative integer *capacity* $cap(e)$ denoting the multiplicity of e (i.e., the number of parallel edges between the two vertices connected by e). We also say that the graph is *capacitated*. E.g., for an edge $e = \{x, y\}$ in an undirected graph we set $cap(e) = 3$ when there are three parallel edges between the two vertices x and y. If the graph is directed, $e = (x, y)$, and $cap(e) = 3$, there are three edges from x to y in G.

Secondly, one can define multigraphs as uncapacitated graphs according to Definition 2.17, but $\mathscr{E}$ is a multiset. By way of example, consider the graph $G = (\mathscr{V}, \mathscr{E})$ with $\mathscr{V} = \{x, y, z\}$ and $\mathscr{E} = \{\{x, y\}, \{x, z\}, \{x, z\}, \{y, z\}\}$. Notice that the edge $\{x, z\}$ appears twice in $\mathscr{E}$, being listed twice.

The following definition introduces a special kind of graphs.

Definition 2.19. *An undirected graph $G = (\mathscr{V}, \mathscr{E})$ is called* bipartite *when $\mathscr{V}$ can be partitioned into two subsets X and Y such that $(e \cap X) \neq \emptyset \neq (e \cap Y)$ holds for each edge $e \in \mathscr{E}$.*

Next, we present some relations between two vertices and between edges and vertices.

Definition 2.20. *Let $G = (\mathscr{V}, \mathscr{E})$ be a multigraph.*

[1] In literature, a simple graph is usually defined without loops and multiple edges. Since we require that there are no loops in a given graph, we can simplify the definition of a simple graph.

(a) *Let G be undirected. Two distinct vertices x and y are said to be* adjacent *if $\{x,y\} \subseteq \mathscr{E}$.* [2] *The* neighborhood *of x is defined by $N(x) := \{y \in \mathscr{V} : \{x,y\} \in \mathscr{E}\}$ contains all vertices adjacent with x (the* neighbors *of x).*

(b) *Suppose that G is a digraph. Then two distinct vertices x and y are adjacent if (x,y) or (y,x) is in $\mathscr{E}$. Let $N^+(x) := \{y \in \mathscr{V} : (x,y) \in \mathscr{E}\}$ and $N^-(x) := \{y \in \mathscr{V} : (y,x) \in \mathscr{E}\}$. We call the vertices in these sets* outneighbors *and* inneighbors*, respectively.*

Note that—even though G might be a graph with multiple edges—$N(x)$ is a set and no multiset (although the neighborhood is sometimes defined as a multiset in literature). E.g., when an edge $\{x,y\}$ occurs four times in G, y appears only once in $N(x)$.

We can also define a relation between vertices and edges.

Definition 2.21. *Let $G = (\mathscr{V}, \mathscr{E})$ be a multigraph.*

(a) *Let G be undirected. We say that an edge e and a vertex x are* incident *if $x \in e$. The multiset of edges incident to x is defined by $\delta(x) := \{e \in \mathscr{E} : e \text{ incident to } x\}$.*

(b) *Given that G is a digraph, x and e are incident if e starts or ends in x, i.e., we define $\delta(x) := \{e \in \mathscr{E} : \exists y \in \mathscr{V} \setminus \{x\} : e = (x,y) \vee e = (y,x)\}$.*

(c) *The* degree *$d(x)$ of a vertex $x \in \mathscr{V}$ is defined as the number of edges incident to x, i.e., $d(x) := |\delta(x)|$.*

(d) *For digraphs, we can define the* indegree *$ind(x)$ and* outdegree *$outd(x)$ of a vertex x as follows:*

$$ind(x) := |N^-(x)|, \quad outd(x) := |N^+(x)|$$

In some applications, edges in a graph are equipped with *edge weights*.

Definition 2.22. *A multigraph is called* weighted *if each edge $e \in \mathscr{E}$ is assigned an integer weight $w(e)$.*

Whenever it holds $w(e) = 1$ for each edge e, we could actually consider an unweighted graph instead. Weights have different applications and may measure the importance or—when nonpositive—the cost of an edge.

Our next definition presents an operation over a vertex subset in digraphs.

Definition 2.23. *Let $G = (\mathscr{V}, \mathscr{E})$ be a digraph. Given a set of vertices $A \subseteq \mathscr{V}$,* merging vertices *in A is defined as the following operation:*

1. *This operation creates a new vertex denoted by $u(A)$;*
2. *for each $w \in \mathscr{V} \setminus A$ such that there is an edge from w to some vertex in A, it creates an edge from w to $u(A)$;*

[2] We point out that $\mathscr{E}$ is a collection of 2-subsets of $\mathscr{V}$. E.g., for $\mathscr{E} = \{\{x,a\},\{y,b\}\}$ (for $x,y,a,b \in \mathscr{V}$, all pairwise different) the edge $\{x,y\}$ does not belong to $\mathscr{E}$ although both the vertices x and y occur in some edge in $\mathscr{E}$.

3. *for each $w \in \mathcal{V} \setminus A$ such that there is an edge from some vertex in A to w, it creates an edge from $u(A)$ to w; and*

4. *it removes all vertices in A and edges incident to them.*

Next, we specify what a directed path in a digraph G is.

Definition 2.24. *Let $G = (\mathcal{V}, \mathcal{E})$ be a digraph.*

(a) *A* directed path *in G is a vertex sequence $(u_1, u_2, \ldots, u_t)$ ($t \in \mathbb{N}$) such that $(u_i, u_{i+1}) \in \mathcal{E}$ for every $i = 1, 2, \ldots, t-1$.*

(b) *A* Hamiltonian path *is a directed path with the additional property that every vertex in the digraph appears exactly once in the path.*

As mentioned in Section 2.1, deciding whether a given graph yields a Hamiltonian path is NP-complete [89].

HAMILTONIAN PATH	
Given:	A simple digraph $G = (\mathcal{V}, \mathcal{E})$.
Question:	Does G contain a Hamiltonian path?

We could also allow the graph to be a multigraph. Nonetheless, we may assume that G is simple since each vertex, and thus each edge, can occur at most once in a Hamiltonian path. We further presume that in a given instance of HAMILTONIAN PATH, each vertex has at least one in- and one outneighbor, i.e., it holds $\min(|N^+(u)|, |N^-(u)|) > 0$ for each vertex u. This is no restriction to our problem because vertices with neither out- nor inneighbor cannot belong to any Hamiltonian path and thus our instance is a NO instance. Provided that a vertex has no outneighbor, this vertex must be the last vertex in a Hamiltonian path (if any) and our instance transforms to a smaller instance with fewer vertices. If there are two or more vertices without any outneighbors, there cannot exist any Hamiltonian path. Likewise, we may argue that vertices without any inneighbor must be placed at the beginning of a Hamiltonian path, provided that one exists.

Another problem tailored to digraphs is the MINIMUM (u,u')-SEPARATOR problem [152] which belongs to P. First of all, let us define a separator.

Definition 2.25. *Let $G = (\mathcal{V}, \mathcal{E})$ be a digraph. For $\mathcal{V}' \subseteq \mathcal{V}$, let $\Gamma_G(\mathcal{V}')$ be the set of all vertices reachable via edges from a vertex in $\mathcal{V}'$, i.e., for every $u \in \Gamma_G(\mathcal{V}')$ there is a directed path from a vertex in $\mathcal{V}'$ to vertex u. For two distinct vertices $u, u' \in \mathcal{V}$, a* (u,u')-separator *is a subset of vertices in $\mathcal{V} \setminus \{u, u'\}$ whose removal destroys all directed paths from u to u'. A minimum (u,u')-separator is a (u,u')-separator with minimum cardinality.*

With these preconsiderations, we define the following problem.

MINIMUM (u,u')-SEPARATOR	
Given:	A digraph $G = (\mathcal{V}, \mathcal{E})$, two designated vertices $u, u' \in \mathcal{V}$ (with $u \neq u'$), and a nonnegative integer ℓ.
Question:	Is it possible to find a (u,u')-separator with cardinality of at most ℓ?

Notice that the definition allows that $(u,u') \in \mathscr{E}$. In this case, there cannot exist any separator since neither u nor u' may be deleted.

Next we regard two other problems related to digraphs which belong to P. The first problem is INTEGRAL FLOW MAXIMIZATION. Our input of this problem is a flow network, defined as follows.

Definition 2.26. *A* flow network *is defined by the tuple (G, cap, x, y), where $G = (\mathscr{V}, \mathscr{E})$ is a simple digraph, $cap : \mathscr{E} \to \mathbb{N}_0$* [3] *a* capacity function*, and x and y are two distinguished vertices. x is called the* source *of the network, whereas y is called the* sink*. The vertices x and y further satisfy the condition $ind(x) = 0 = outd(y)$.*

The following definition specifies what a flow is.

Definition 2.27. *Let (G, cap, x, y) be a flow network defined as in Definition 2.26. An* integral flow *F is defined as a mapping $F : \mathscr{E} \to \mathbb{N}_0$ with the following properties:*

1. $\sum_{e \in \delta(x)} F(e) = \sum_{e \in \delta(y)} F(e) =: \hat{s}$.
2. $\sum_{z \in N^-(w)} F((z,w)) = \sum_{z \in N^+(w)} F((w,z)) (= \hat{s})$ *holds for each* $w \in \mathscr{V} \setminus \{x,y\}$.
3. $F(e) \leq cap(e)$ *holds for each* $e \in \mathscr{E}$.

We refer to the first two conditions as *flow conservation conditions*, the third constraint is to ensure that capacities are met by all edges in $\mathscr{E}$. An integral flow F satisfying all three conditions is said to be *feasible*. $\hat{s}$ is called the *flow size* or *flow value* of F. Now we are equipped to define the integral flow maximization problem which is known to be in P [1].

INTEGRAL FLOW MAXIMIZATION	
Given:	A flow network defined as in Definition 2.26 and a nonnegative integer $\hat{s}$.
Question:	Can we find a feasible integral flow F (that is, a mapping $F : \mathscr{E} \to \mathbb{N}_0$ satisfying the three conditions in Definition 2.27) such that the flow size of F is at least $\hat{s}$?

Flow networks apply to many problems, such as assignment problems (cf. [1]). Intuitively, a flow network with a flow of size $\hat{s}$ corresponds to a pipeline system flown through by water volume $\hat{s}$. An example of a flow network is given in Figure 2.2.

There is yet another problem about flow networks that is in P [1]:

INTEGRAL MIN-COST FLOW	
Given:	A flow network defined as in Definition 2.26, a nonnegative integer $c(e) \in \mathbb{N}_0$ for each $e \in \mathscr{E}$ (*edge costs*), and two nonnegative integers $\hat{c}, \hat{s}$.
Question:	Can we find a feasible integral flow F with flow size $\hat{s}$ such that $\sum_{e \in \mathscr{E}} c(e)F(e) \leq \hat{c}$?

In other words, we ask whether there is a flow of fixed size $\hat{s}$ such that the total cost of the flow does not exceed a certain threshold $\hat{c}$.

A class of problems tailored to undirected graphs is defined next. The most general of these problems can be found in [9, 101] and is known to be tractable (for membership in P, see also [88]).

[3] A common flow network has a real-valued capacity function *cap*, but in our applications the edge capacities are always nonnegative integers.

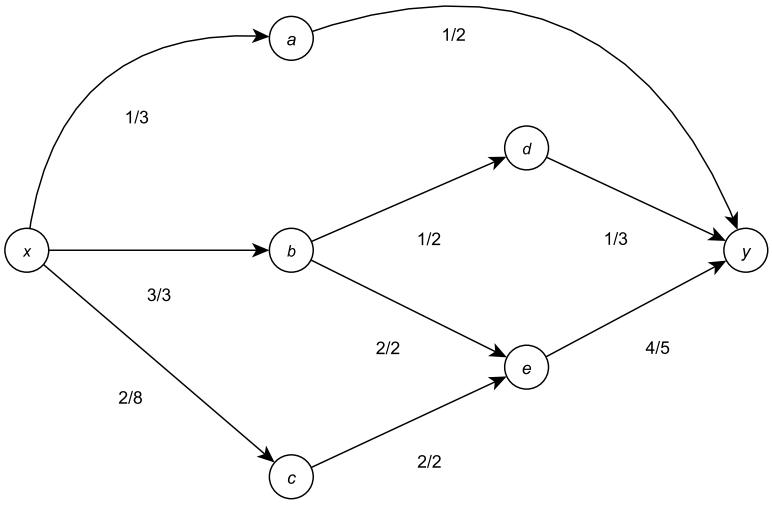

Figure 2.2: An example of a flow network with source x and sink y. Each edge is labeled with two numbers. The left number corresponds to the number of units assigned by the feasible flow F to this edge, whereas the right number represents the capacity of the edge. Observe that the flow is integral, feasible, and has value $\hat{s} = 6$. Nonetheless, the flow is not maximum as one could shoot one further flow unit through the edges (x,a) and (a,y).

GENERALIZED b-EDGE MATCHING	
Given:	An undirected capacitated simple graph $G = (\mathscr{V}, \mathscr{E})$, capacity functions (or "degree bounds") $b_l, b_u : \mathscr{V} \to \mathbb{N}_0 \cup \{\infty\}$, edge capacity function $cap : \mathscr{E} \to \mathbb{N}_0$, and a nonnegative integer μ.
Question:	Can we find a function $\nu : \mathscr{E} \to \mathbb{N}_0$ with $\sum_{e \in \mathscr{E}} \nu(e) \geq \mu$ such that $(*)$ $b_l(y) \leq \sum_{e \in \mathscr{E} \cap \delta(y)} \nu(e) \leq b_u(y)$ holds for each vertex $y \in \mathscr{V}$ and $(**)$ $\nu(e) \leq cap(e)$ holds for each $e \in \mathscr{E}$?

A collection of edges or, more precisely, a mapping ν satisfying $(*)$ and $(**)$ is called a *feasible matching* (or *matching*, for short). In particular, feasibility means that no vertex or edge capacity constraint is hurt. The mapping ν assigns to each edge e the number of times e occurs in the matching. E.g., given $\nu(e) = 3$, edge e occurs thrice is the matching. $\nu(e) = 0$ means that edge e is not in the matching. Observe that the underlying graph G is formally simple and $\mathscr{E}$ and $\delta(y)$ ($y \in \mathscr{V}$) are proper sets and no multisets. Since G is endowed with an edge capacity function, in fact we are dealing with a multigraph.

The problem is sometimes also called GENERAL b-MATCHING or GENERAL b-EDGE MATCHING in literature. Note that the original definitions in [9] and [101] are even held more general in a sense that the graph may contain loops, general lower edge constraints, the edges may be equipped with weights, the capacities and weights may be arbitrarily real-valued, or the matching itself may contain negative numbers. In our setting, it suffices to focus on the unweighted version (or, more precisely, the weights are constant), nonnegative capacities, and hence (due to conditions $(*)$ and $(**)$) on a nonnegative matching mapping. This makes sense as each edge cannot be selected a

negative number of times in practice. Moreover, we do not need non-zero lower edge capacities.

We obtain the definition of GENERALIZED b-EDGE COVER by replacing the expression $\sum_{e\in\mathscr{E}} \nu(e) \geq \mu$ with $\sum_{e\in\mathscr{E}} \nu(e) \leq \mu$. In the former case, we want to maximize the number of edges, while in the latter case we try to minimize the number of edges.

Important special cases of GENERALIZED b-EDGE MATCHING and GENERALIZED b-EDGE COVER are b-EDGE MATCHING and b-EDGE COVER, respectively. For the matching problem, we have $b_l(y) = 0$ for each $y \in \mathscr{V}$. In contrast, we let $b_u(y) = \infty$ for the cover problem.

For practical reasons, we denote the lower capacity restrictions with $b(y)$ ($y \in \mathscr{V}$) instead of $b_l(y)$ when dealing with (standard) b-edge cover (that is, there are no upper capacity restrictions). Likewise, for (standard) b-edge matching, we denote the upper capacities with $b(y)$ instead of $b_u(y)$.

For the sake of simplicity, we say *generalized b-edge matching* and *generalized b-edge cover* and use these notions interchangeably with *(b-)edge matching* and *(b-)edge cover*, respectively, when there is no way of confusion.

One of the most special matching problems in this thesis is the problem of finding a perfect matching in a simple bipartite graph.

PERFECT MATCHING FOR BIPARTITE GRAPHS	
Given:	A simple bipartite graph $G = (\mathscr{V} = X \dot\cup Y, \mathscr{E})$.
Question:	Can we find a *perfect matching*, that is, a subset $\mathscr{E}' \subseteq \mathscr{E}$ such that every vertex in $\mathscr{V}$ occurs in exactly one edge in $\mathscr{E}'$?

Note that a perfect matching can only exist for $|X| = |Y|$. While PERFECT MATCHING FOR BIPARTITE GRAPHS is in P, counting the number of perfect matchings for simple bipartite graphs is #*P*-complete [157]. In other words, PERFECT MATCHING FOR BIPARTITE GRAPHS is easy to decide, but hard to count. The counting version is defined as follows:

#PERFECT MATCHINGS FOR BIPARTITE GRAPHS	
Given:	A simple bipartite graph $G = (\mathscr{V} = X \dot\cup Y, \mathscr{E})$.
Question:	How many perfect matchings does G have?

For practical reasons, we also say #PERFECT MATCHINGS instead of #PERFECT MATCHINGS FOR BIPARTITE GRAPHS. Note that counting the number of perfect matchings is also hard for general simple graphs (i.e., not bipartite) as a bipartite graph is a special case of a general graph.

We finally point out that the definition of generalized matching is tailored to capacitated graphs (or, more precisely, to uncapacitated simple graphs that become capacitated due to the capacity constraints $cap(e)$). In our proofs, we follow the arguing in the largest part of the COMSOC literature and argue by means of uncapacitated multigraphs. In other words, parallel edges between the same two vertices are considered as separate, distinct edges with upper "capacity" one each. [4] The matching mapping ν maps from $\mathscr{E}$ to $\{0,1\}$ instead of $\mathbb{N}_0$ and both $\mathscr{E}$ and $\delta(y)$ ($y \in \mathscr{V}$) are multisets in this variant. We can further do without edge capacities $cap(e)$, when the underlying graph is a multigraph, as these capacities would be trivially met by each matching, due to $\nu(e) \in \{0,1\}$ and

[4] We use the quotation marks as the multigraph representation actually gets along without edge capacities.

$cap(e) = 1$ for each e in the edge multiset $\mathscr{E}$. It holds $\nu(e) = 1$ or $\nu(e) = 0$, depending on whether or not an edge e is in the matching.

Working with multigraphs in our proofs makes things easier when edges one-to-one correspond to voters the more so as arguing with multiplicities of edges is more cumbersome than arguing with single voters/edges. Nevertheless, whensoever we define a multigraph in a proof, we immediately obtain a corresponding unique capacitated simple graph as in the definition of generalized b-edge matching: for each pair of distinct vertices x and y, we simply define $cap(\{x,y\})$ as the number of edges between x and y in the multiset $\mathscr{E}$. Moreover, all parallel edges in a multigraph between the same two vertices x and y can be unified by one edge representing these edges (a similar arguing can be found in [119]; for capacitated graphs, we also refer to [151]). Loosely speaking, in our proofs we define an uncapacitated multigraph and actually compute a matching/cover for the corresponding capacitated graph.

2.3 Introduction to Voting Theory

In this section, we give a short introduction to voting theory. For the interested reader, we also refer to the textbooks [28] and [146] for further reading. Let us begin with the most basic notions.

2.3.1 Formal Background

Formally, an *election* is a pair (C,V), where C is a set of m *candidates* and V is a multiset of n *voters*, where $m \in \mathbb{N}$ and $n \in \mathbb{N}_0$, unless stated otherwise. [5] We will use the notions *candidate* and *alternative* interchangeably. In canonical settings, we have $C = \{c, c_1, \ldots, c_{m-1}\}$ and $V = \{v_1, \ldots, v_n\}$. c is also called *distinguished candidate* or *designated candidate*. When studying a destructive problem, c is also denoted by *despised candidate*. We refer to all other candidates—canonically the c_j—as *non-distinguished candidates*. In some applications, we rename candidates and/or voters depending on the context. For example, we sometimes write small letters such as a, b, d, or e to denote general candidates (possibly c is among them) or non-distinguished candidates. At times, we also denote voters by v or even by u or w.

Each voter $v_i \in V$ $(i = 1, \ldots, n)$ is represented via his *preference order* $\succ_i$ (or $\succ_{v_i}$) over the candidates in C which is a *strict linear order* in the standard setting. When the voter is clear from the context, we also say $a \succ b$ instead of $a \succ_i b$ or $a \succ_{v_i} b$ for two given candidates a and b. A strict linear order is a binary relation, that is total, asymmetric and transitive. We use the terms *ranking* and *preference order* interchangeably. Likewise, a *complete ranking* is the same as a *strict linear order*. We often drop the word "strict" as voters are canonically defined as linear orders without ties. Intuitively, a linear order ranks the candidates from first to last. For example, $a \succ_i b \succ_i c$ means that voter v_i prefers a to b, b to c, and—due to transitivity—a to c. For disjoint candidate subsets A and B with $a \notin A \cup B$, $A \succ_v a \succ_v B$ means that voter v prefers all candidates in A to a, a to all candidates in B, and—due to transitivity—all candidates in A to all candidates in B. Whether

[5] One could also set m as a nonnegative integer, but we consider only elections containing at least a distinguished candidate c. By contrast, elections without any voters may make sense. For instance, it may occur that a candidate is only a (necessary) winner when there are no voters or that a chair can only make his favorite candidate a winner by deleting all voters. In Chapter 5, we provide some polynomial-time algorithms where—depending on the chair's resources—an election with empty voter set may be the result.

and how the candidates in A or B are ranked among themselves, respectively, does not matter. When we write $a \succ_v A$ for a candidate subset A and candidate a and we know that v is a complete vote, we implicitly assume that the candidates in A are ranked according to an arbitrary, but fixed ordering.

By $pos(d,v)$, we denote the position of a candidate d in a voter v's ranking (when the candidate set C is clear from the context). Likewise, $pos_{\tilde{C}}(d,v)$ indicates the position of candidate d in vote v when referring to the candidates in $\tilde{C}$.

We refer to $\{v_1,\ldots,v_n\}$ as an *n-voter profile* P. A profile P is said to be a *complete profile* when all votes v_i are complete. Note that we can restrict preference profiles to all subsets and supersets of C or V. Intuitively, we may assume a maximal (in the sense of non-extendable) and universal election $(\hat{C},\hat{V})$ where each voter $v \in \hat{V}$ declares a complete ranking over all candidates in $\hat{C}$. As $V \subseteq \hat{V}$ and $C \subseteq \hat{C}$, the "universal" rankings can be restricted only to the voters in V and merely to the candidates in C, by erasing all irrelevant voters and candidates. Formally, an election (C,V) is a projection of the election $(\hat{C},\hat{V})$ onto the restricted election (C,V). Analogously, we can extend or restrict the rankings of election (C,V) to arbitrary (other) elections (C'',V''), where C'' and V'' are any subsets of $\hat{C}$ and $\hat{V}$, respectively. This assumption will turn out useful when considering control by adding, deleting, or replacing candidates and/or voters. As an example, let $\hat{C} = \{c_1,\ldots,c_8\}$ and $\hat{V} = \{v_1,v_2,v_3\}$, where $v_1,v_2 : c_1 \succ c_2 \succ \ldots \succ c_7 \succ c_8$ and $v_3 : c_8 \succ c_7 \succ \ldots \succ c_2 \succ c_1$. Let $C = \{c_1,c_4,c_6\}$ and $V = \{v_1,v_3\}$. Then the election (C,V) is defined by the candidate set C, voter set V, and the rankings $v_1 : c_1 \succ c_4 \succ c_6$ and $v_3 : c_6 \succ c_4 \succ c_1$.

A *voting rule* $\mathscr{F}$ maps an election to a subset of candidates we refer to as the *winners* of the election. More precisely, a voting rule is defined as a set-valued function $\mathscr{F} : (C,V) \to \mathscr{P}(C)$ and the candidates in $\mathscr{F}(C,V)$ are the winners of the election. We point out that in literature a *voting rule* frequently requires that $|\mathscr{F}(C,V)| = 1$ for each input election (C,V). Hence, we say *voting rule* while actually regarding *voting correspondences* mapping an election to $\mathscr{P}(C)$. Note that one can also define social welfare functions that output—given an election—a complete social ranking over all alternatives. Social welfare functions are not studied in this thesis.

As the different definitions suggest, a voting rule may yield an arbitrary number of winners, depending on which voting rule and which winner model we regard. According to the *co-winner model*, a given candidate c is a winner of a given election if and only if c is either one among several winners or the only winner of the election. Formally, it holds $c \in \mathscr{F}(C,V)$. Under the *unique-winner model*, c is a winner of the election if and only if c is the only winner, or formally, $\mathscr{F}(C,V) = \{c\}$. We will find that sometimes the co-winner model and the unique-winner model lead to different complexities for one and the same problem. In the following, we will merely study so-called *single-winner voting rules* where one winner is sought (although there may be several winners under the co-winner model). In particular, we will not consider *multi-winner voting rules* where a committee of fixed size s ($s > 1$) has to be found (see i.a. [53]).

There are different ways to define the winners of an election as we will see in Section 2.3.2. For example, several voting rules assign a score to each candidate. According to the co-winner model, all candidates with the maximum score (i.e., there is no candidate with a higher score) are the winners. In the unique-winner model, either a unique candidate has the maximum score and wins or there is no winner at all. There are many other ways to determine the winners of an election. Once again, we refer to the textbooks [146, 28]. Both winner models make sense in practice. For example, in political elections only one president can be elected. In other applications, the number of winners may be arbitrary. Suppose for instance that in a (simplified) political election each voter

votes for his favorite party and all parties reaching at least a certain threshold of votes make it to the parliament (for example, a party's vote must exceed the five-percent threshold). All parties in the parliament are declared as the winners. Although this example also applies to multi-winner voting rules, there might be one winner or even no winner at all (and revoting in practice). Since there is no restriction [6] on the number of parties in the parliament, the co-winner model is more suitable than the unique-winner model in this context. Another example arises from rankings of whatever kind, such as the best goal scorers ever in football world championships. All best goal scorers are the "winners" of this ranking (one can interpret each shot goal as a voter and the players as candidates; the candidates with the highest score are the winners). [7] Note that in case a candidate is a winner under the unique-winner model (we say *the* winner of the election), the candidate is a winner under the co-winner model as well. The opposite direction does not hold in general.

We point out that in some settings a *tie-breaking rule* is used whenever at least two candidates are tied for the victory. Various tie-breaking rules are thinkable and known from literature. For example, a tie-breaking rule can always select the smallest or largest candidate depending on a predetermined lexicographical order (e.g., the alphabetical order of the candidates' names or the age of the candidates). Note that such tie-breaking schemes are not neutral. Loosely speaking, a voting rule (together with a tie-breaking rule) is *neutral* when all candidates are treated equally and permuting the candidates' names changes the winner(s) of the election accordingly. Other tie-breaking rules may draw a voter (predetermined or at random, cf. [5]) and the tying winner ranked best possible by this voter is declared as the election winner. Such tie-breaking rules—or, more precisely, voting rules equipped with such tie-breaking rules—are neutral, but not *anonymous* in general, in a sense that permuting the names of the voters and leaving their rankings unchanged may yet change the winner. Other tie-breaking rules satisfying both fairness conditions might take some qualitative properties of the candidates into account. As an example, a tie-breaking rule selects the candidate with the largest number of voters ranking this candidate first or with the smallest number of voters ranking him last. Note that such tie-breaking rules must still be refined as there may be still ties after applying the tie-breaking rule. We refrain from using tie-breaking rules by and large. One exception is in Chapter 5 where we study the two voting rules Plurality with Runoff and Veto with Runoff. For these rules, we do not assume any particular tie-breaking method, but apply a generalization of the co-winner model to the original election in a sense that we ask whether there is at least one way to break ties such that the distinguished candidate c and another candidate reach the runoff and c is a winner in the runoff. This model is called *parallel-universe tie-breaking* in literature. In summary, tie-breaking rules are often reasonable in practice because in many applications (such as electing a committee of fized size, the dean of a faculty, or the president of a football club) it is a desirable property of a voting rule to determine a unique winner. Such voting rules are also called *resolute*.

In the following section, we present a variety of voting rules.

2.3.2 Voting Rules

There are various voting rules known from literature. Our main focus lies on *scoring rules*. Each scoring rule is defined by a vector $\sigma = (\sigma_1, \ldots, \sigma_m)$ (with $m = |C|$) for which $\sigma \in \mathbb{N}_0^m$ and $\sigma_i \geq \sigma_j$ for each $1 \leq i \leq j \leq m$. Each voter assigns σ_1 points to his favorite candidate, σ_2 to his second

[6] Effectively, there can be at most twenty parties reaching or exceeding the five-percent threshold.

[7] The voting rule used in this example is the Plurality rule defined in the next section.

most preferred candidate etc., and σ_m points to his least preferred alternative. The candidates with the highest score are the winners under the co-winner model. The scoring rules introduced in the following belong to the most prominent scoring rules.

- k-Approval ($k \in \mathbb{N}$) is defined by the scoring vector $\sigma = (\underbrace{1,\ldots,1}_{k},0,\ldots,0)$. In k-Approval, each voter gives one point to his k most favorite candidates and zero points to all remaining candidates. Formally, it holds $\sigma_i = 1$ ($i \in [k]$) and $\sigma_i = 0$ ($i > k$). 1-Approval is also known as the Plurality rule.

- k-Veto ($k \in \mathbb{N}$) is defined by the scoring vector $\sigma = (1,\ldots,1,\underbrace{0,\ldots,0}_{k})$. In k-Veto, each voter assigns zero points to his k least preferred alternatives and one point to all other candidates. We say Veto instead of 1-Veto. For a fixed number m of candidates, k-Approval is equivalent to $(m-k)$-Veto. Note that the voting rules do not coincide for variable m as we assume k to be a constant, unless stated otherwise.

  In several proofs, we argue with veto numbers instead of approval scores. We say that a voter vetoes candidate d if and only if he assigns zero points to d. Otherwise, the voter does not veto d. It always holds $\mathit{vetoes}(d) + \mathit{score}(d) = |V|$, where $\mathit{score}(d)$ denotes the score of d according to the scoring vector of the k-Veto rule and $\mathit{vetoes}(d)$ is the number of voters assigning zero points to d.

- The Borda rule is defined by the scoring vector $(\sigma_1,\ldots,\sigma_m)$ where $\sigma_i = m - i$ ($1 \leq i \leq m$). In other words, a voter's top candidate gets $m-1$ points, his second most preferred candidate $m-2$ points, and so on. [8] Accordingly, a voter's least preferred candidate receives no point from this voter. We point out that a voting rule, using a scoring vector similar to the one defining the Borda rule, is used at the Eurovision Song Contest. This adaption however assigns slightly different scores to the candidates and is a truncated version, in a sense that about half of all candidates do not receive any point.

Note that there are infinitely many scoring vectors. Our focus lies on k-Approval and k-Veto.

Aside from scoring rules, there are voting rules based on pairwise comparisons. For two distinct candidates a and b, we introduce $N(a,b)$ as the number of voters preferring a to b. Formally, for an election (C,V) and $a,b \in C$, $a \neq b$ (which requires that $|C| > 1$), we define

$$N(a,b) := |\{v \in V : \quad a \succ_v b\}|.$$

We say that a wins the *pairwise comparison* (similar expressions are *pairwise election, pairwise contest*, or *head-to-head contest*) against b if and only if $N(a,b) > \frac{|V|}{2}$. Moreover, a loses the pairwise comparison against b if and only if $N(a,b) < \frac{|V|}{2}$. We further say that a ties with b if

[8] In literature, the Borda rule is often defined by the scoring vector $\sigma = (m, m-1, m-2, \ldots, 2, 1)$. We follow the predominant convention that a voter's least preferred candidate should get zero points from this voter. Nevertheless, both rules are equivalent in a sense that they output the same winner sets for all input instances. This can be verified as follows: Each candidate d has $\mathit{score}(d)$ points according to the Borda variant presented in this section and $\mathit{score}(d) + |V|$ points according to the other variant. Hence, $\mathit{score}(d)$ is maximum among all candidates if and only if $\mathit{score}(d) + |V|$ is maximum among all candidates.

and only if $N(a,b) = \frac{|V|}{2} (= N(b,a))$. Besides, a does not lose the pairwise comparison against b or, equivalently, at least ties with b if and only if $N(a,b) \geq \frac{|V|}{2}$. Note that for odd numbers of voters two candidates never tie with each other. One of the most prominent rules based on pairwise comparisons is the Condorcet rule. A candidate $a \in C$ is a Condorcet winner if and only if $N(a,b) > \frac{|V|}{2}$ holds for each candidate $b \in C \setminus \{a\}$. In other words, a Condorcet winner (if one exists) wins the head-to-head contests against all other candidates. In case such a candidate does not exist, the Condorcet rule outputs an empty winner set. Similarly, all candidates not losing any pairwise comparison are the winners according to the Weak Condorcet rule. Formally, a is a *weak Condorcet* winner if and only if it holds $N(a,b) \geq \frac{|V|}{2}$ for each $b \in C \setminus \{a\}$. While each election yields at most one Condorcet winner, there can exist an arbitrary number of winners according to the Weak Condorcet rule. Note that for odd numbers of voters the two voting rules coincide.

There are further, more or less constructed rules based on pairwise comparisons, such as Maximin, Copeland, Ranked Pairs, Cup, or Dodgson. Since they are immaterial to our analysis, we refer the interested reader to [156] (for Ranked Pairs) and [146].

We say that a voting rule $\mathscr{F}$ is *Condorcet consistent* if the Condorcet winner—provided that one exists—is always a winner according to $\mathscr{F}$. Like neutrality and anonymity, Condorcet consistency may be considered as a desirable property of a voting rule. We point out that scoring rules are not Condorcet consistent as the Condorcet winner need not be among the winners in general. As an example, the Plurality rule may have the Condorcet loser (the candidate losing all pairwise comparisons) as its unique winner (e.g., suppose that two voters favor c and three voters favor a, b, and d, respectively, and all of them rank c last) [9] or the Condorcet winner is the Plurality loser (assume that there are five voters, each voter ranks c second, the candidates a, b, d, e, and f are ranked first by exactly one voter each, and we have $C = \{a,b,c,d,e,f\}$). The Borda rule guarantees at least that the Condorcet loser is not a Borda winner although the Condorcet winner is not a Borda winner in general [146].

We consider two further voting rules: Plurality with Runoff and Veto with Runoff which can be regarded as hybridizations of scoring rules and voting rules based on pairwise comparisons. The Plurality with Runoff rule is defined as follows.

1. In the preround, we compute the Plurality scores $score(d)$ for every candidate $d \in C$.

2. The two candidates with the highest scores move to the runoff stage. In case of ties, a tie-breaking rule must be applied to uniquely determine the candidates moving to the runoff. Assume that a and b are the two runoff candidates. Both under the co- and unique-winner model, a runoff winner is a candidate in $\{a,b\}$ preferred by a majority of voters over the other candidate in the runoff. If $N(a,b) = N(b,a)$ holds, both candidates are the winners in the co-winner model, whereas there is no winner under the unique-winner model. Formally, we apply the Weak Condorcet (Condorcet) rule to the original election restricted to the two runoff candidates when the underlying winner model is the co-winner model (unique-winner model). As mentioned above, we will sometimes assume that ties are broken in favor of the distinguished candidate, both in the preround and in the final runoff. We point out that there are variants of Plurality with Runoff in literature where each voter may revote in the runoff election. This is often the case in political elections. Accordingly, it might be possible that

[9] This paradox is also known under the name *Borda Paradox*

Voting rule	a	b	c	d	e
Plurality	2	1	0	2	1
2-Approval	2	1	1	3	5
3-Approval	3	1	5	4	5
Veto	3	0	1	1	1
Borda	10	9	11	14	16

Table 2.1: Scores of the election (C,V) in Example 2.28 for different scoring rules.

a voter prefers a to b in his initial ballot, but in the runoff election, he suddenly changes his opinion to $b \succ a$. Our version of Plurality with Runoff forbids such revoting.

The Veto with Runoff rule is defined analogously, but with the difference that the two candidates with the fewest veto numbers move to runoff. Again, it may be necessary that a tie-breaking rule determines the two candidates moving to the runoff stage.

Observe that both rules require two rounds until a decision is reached.

Last but not least, we introduce the Approval rule. Where all voting rules mentioned up to now require that all voters specify complete rankings over the set of candidates, given the Approval rule each voter provides a $\{0,1\}$-vector with $m = |C|$ components. In the vector representing voter v's preferences, the j-th component is one (zero) if and only if v approves of (disapproves of) the jth candidate. The candidate(s) with the highest score is (are) the winner(s). Note that, although we count the approval scores for all candidates, the Approval rule does not belong to the classical scoring rules since each voter may approve of arbitrary numbers of candidates and therefore the scoring vector is not unique in general and may differ from voter to voter. We refrain from studying the Approval rule, but have a closer look at the related voting rule k-Approval requiring each voter to approve of exactly k candidates.

The following example shall illustrate how the different voting rules described so far work.

Example 2.28. *Let (C,V) be an election with candidate set $C = \{a,b,c,d,e\}$, voter set $V = \{v_1,\ldots,v_6\}$, and the following voters' rankings:*

$$\begin{aligned} v_1 &: \quad a \succ e \succ d \succ b \succ c \\ v_2 &: \quad a \succ e \succ c \succ b \succ d \\ v_3 &: \quad d \succ c \succ a \succ b \succ e \\ v_4 &: \quad d \succ e \succ c \succ b \succ a \\ v_5 &: \quad e \succ d \succ c \succ b \succ a \\ v_6 &: \quad b \succ e \succ c \succ d \succ a \end{aligned}$$

We next determine the winners of election (C,V) for various voting rules. Let us start with scoring rules. Table 2.1 presents the scores according to the different scoring rules. The winners (in the co-winner model) are marked red. For Veto, we compute the veto numbers instead of scores. Note that $vetoes(c') + score(c') = |V| = 6$ for each candidate $c' \in C$, where $score(c')$ counts the number of voters not vetoing c' (i.e., giving one point to c').

For Borda, each voter's top choice is awarded four points, the second preferred candidate of a voter gets three points, and so on, and each voter's least preferred alternative receives no point.

Candidate pair (c',c'')	(a,b)	(a,c)	(a,d)	(a,e)	(b,c)	(b,d)	(b,e)	(c,d)	(c,e)	(d,e)
$N(c',c'') : N(c'',c')$	3:3	2:4	2:4	3:3	2:4	2:4	2:4	2:4	1:5	2:4
Winner(s) of comparison	tie	c	d	tie	c	d	e	d	e	e

Table 2.2: Head-to-head contests for the election (C,V) in Example 2.28.

By way of example, we compute $score(a) = 4+4+2+0+0+0 = 10$ as the Borda score for candidate a.

For scoring rules, we point out that all five candidates are winners for at least one rule studied. Hence, the selection of the appropriate voting rule may turn out to be a difficult task in practice. For example, the Plurality rule selects a as a winner (together with d) although half of all voters consider a as their worst choice. Moreover, d and a are treated equally by the Plurality rule although d seems to be by far a more balanced candidate than a, for d is ranked first by two voters and ranks once on all other positions each. In contrast, five out of six voters rank e on first or second place, but the Plurality rule yet considers a as a better candidate than e. Observe that the Plurality rule might tend to select polarizing candidates.

In contrast, the 3-Approval rule selects c and e (in the co-winner model); both candidates are ranked third or better by five of six voters.

In opposition, the Veto rule uniquely decides that b is the winner, being the only candidate without any vetoes. Hence, the Veto rule seems appropriate when an election designer wants to minimize the number of totally unsatisfied voters (when we assume a voter to be "totally unsatisfied" in case his least preferred alternative is elected by the voting rule).

The Borda rule appears to be the most "honest" rule due to the fact that it awards points according to each position in a voter's vote and therefore reflects all information provided by the voters. Note that this is true only to a certain extent: Possibly, a voter perceives the distance between his favorite candidate and his second most preferred candidate as much greater than the distance between any other two adjacent candidates in his ranking. Such preferences could be better modeled by giving the voters the possibilities to assign individual scores to every candidate. What we can say is that the Borda rule better approximates the voters' preferences than k-Approval or k-Veto by exploiting more information given by the voters' rankings. In contrast, Plurality and Veto treat all candidates not ranked first or last equally. Despite all the advantages of the Borda rule compared to k-Approval or k-Veto, the Borda rule seems impracticable for large elections with many candidates as voters might be unwilling to provide complete rankings over dozens of candidates (Emerson [59] made several suggestions how the Borda rule can be applied to partial ballots in a sense that voters need not rank all candidates).

Next let us focus on Condorcet and Weak Condorcet. Table 2.2 provides an overview of the head-to-head contests between all pairs of distinct candidates.

Observe that there is no Condorcet winner as no candidate wins all pairwise comparisons. However, e is a weak Condorcet winner since e wins the head-to-head contest against b, c, and d, and ties with a. All other candidates lose at least one pairwise election and are thus no winners according to the Weak Condorcet rule.

Under the Plurality with Runoff rule, a and d move to the runoff with two points each and d finally beats a in their head-to-head contest. Given the Veto with Runoff rule, b and one of the candidates c, d, and e reach the runoff and b loses against the other candidate in the runoff, regardless

of which candidate in $\{c,d,e\}$ actually makes it to the runoff.

All in all, we realize that the selection of the best voting rule to reach a decision may induce various discussions and each voting rule itself has advantages as well as disadvantages.

2.3.3 Problem Settings

In this section, we define several problems in and around elections. Many of these problems and some variants of them are studied in Chapter 3 and the subsequent chapters. In most of our problem definitions throughout this section, our input contains an election (C,V). If not mentioned other, we implicitly assume that C is a set of $m \in \mathbb{N}$ candidates and V is a multiset of $n \in \mathbb{N}_0$ complete votes. For multimode control, our input is an election $(C \cup D, V \cup W)$ with a set C of registered candidates, a set D of unregistered candidates, a set V of registered voters, a set W of unregistered voters (all four sets are finite), and each voter in $V \cup W$ specifies a strict linear order over all candidates in $C \cup D$. We do not specify the underlying winner model since all definitions can be easily adapted to the co-winner model, unique-winner model, or to the unique-winner model combined with some tie-breaking rule.

We start with the definition of the *Winner* problem which possibly belongs to the most basic problems in voting theory. This problem goes back to [15].

$\mathscr{F}$-WINNER	
Given:	An election (C,V) and a designated candidate $c \in C$.
Question:	Is c a winner of the election (C,V) under voting rule $\mathscr{F}$?

We point out that the winner problem is easy for most voting rules, especially those studied by us. Given a scoring rule, we just have to compute the scores and compare afterwards which candidates have the highest score. Similarly, we can easily check whether a candidate is a Condorcet or a weak Condorcet winner as for each pair (a,b) of distinct candidates, we compute the values $N(a,b)$. Note that we actually have to calculate $\frac{m(m-1)}{2}$ values since we can exploit the condition $N(b,a) = |V| - N(a,b)$.

Some few somewhat constructed voting rules have a hard winner problem, such as Dodgson or Kemeny [15].

Unfortunately, there are different ways how an external agent can influence the outcome of an election. By way of example, a briber can alter some voters' votes in order to make a given candidate win the election. *Bribery* was introduced in [78, 79] and is defined as follows:

$\mathscr{F}$-BRIBERY	
Given:	An election (C,V), a designated candidate $c \in C$, and a nonnegative integer ℓ (the *bribery limit*).
Question:	Is it possible to make c a winner of the election (C,V) under $\mathscr{F}$ by changing at most ℓ votes?

We also say $\mathscr{F}$-CONSTRUCTIVE BRIBERY as the briber has a "constructive" goal. The problem $\mathscr{F}$-DESTRUCTIVE BRIBERY was introduced by Xia [165] and asks whether or not a despised candidate can be prevented from winning. More precisely, Xia studied the problem *Margin of Victory* which asks how many voters must change their rankings such that the current winner changes.

We often denote the final election with $(C,\bar{V})$. Note that the voters' names remain unchanged when a briber bribes some voters, but the rankings of the bribed voters differ from their original rankings in general. Observe that $(C,\bar{V})$ and (C,V) are identical concerning the votes not bribed.

Faliszewski et al. [78, 79] also studied more general models where for each voter v_i a weight w_i and a price tag p_i are given. The value p_i indicates the amount the briber has to pay to voter v_i in order that v_i changes his vote according to the briber's wishes. In case $p_i = \infty$, voter v_i is unbribable. Moreover, w_i represents the weight of voter v_i's vote. E.g., when $w_i = 3$, the vote of v_i counts as three votes (of weight one). In the priced variant, the briber must not exceed the bribery budget ℓ. The problem variation in the definition of $\mathscr{F}$-BRIBERY is a special case of weighted bribery with price tags where $p_i = w_i = 1$ holds for each voter v_i. We restrict ourselves to this standard variant the more so as many problems in the weighted and/or priced variant are hard even under full information (cf. [79, 119]). This standard variant has been studied by many researchers since the original papers by Faliszewski et al. [78, 79]. Moreover, the standard case is suitable when we do not know the voters' price tags in advance, when the voters have nearly identical prices, or when we just want to know how many voters have to be bribed.

Another way to tamper with the outcome of an election is given by *Electoral Control*. Our focus lies on these control variants where an external agent—the *chair*—may add or delete some voters or candidates to change the outcome of the election. One of the most general of these problems is *Multimode Control* [81], defined as follows.

$\mathscr{F}$-MULTIMODE CONTROL	
Given:	An election $(C \dot\cup D, V \dot\cup W)$ with registered candidate set C, unregistered candidate set D, registered voter set V, unregistered voter set W, a designated candidate $c \in C$, and four non-negative integers $\ell_{AV}, \ell_{DV}, \ell_{AC}, \ell_{DC}$, with $\ell_{AV} \leq \|W\|$, $\ell_{DV} \leq \|V\|$, $\ell_{AC} \leq \|D\|$, and $\ell_{DC} \leq \|C\|$.
Question:	Are there subsets $V' \subseteq V$, $W' \subseteq W$, $C' \subseteq C$, and $D' \subseteq D$, with $\|V'\| \leq \ell_{DV}$, $\|W'\| \leq \ell_{AV}$, $\|C'\| \leq \ell_{DC}$, and $\|D'\| \leq \ell_{AC}$, such that c is a winner of the election $((C \setminus C') \cup D', (V \setminus V') \cup W')$ under voting rule $\mathscr{F}$?

We obtain the definition of $\mathscr{F}$-DESTRUCTIVE MULTIMODE CONTROL by asking whether c can be prevented from being a winner. In order that the problem does not become trivial, we have to add the constraint that $c \notin C'$ in the destructive version. Otherwise, the chair reaches his goal simply by deleting the despised candidate c. We could also require that $\ell_{DC} < \|C\|$ in the constructive version as a reasonable chair never deletes his favorite candidate. Moreover, we point out that in the original definition of multimode control in [81] the chair may additionally bribe some voters. By contrast, we study control and bribery separately from each other. The reasons are as follows. On the one hand, we follow the largest part of research strictly differentiating between bribery and control. On the other hand, combining multimode control and bribery would entail various additional technical requirements. So it is unclear whether the briber may bribe deleted or added voters. Similar other technical questions must still be clarified. Nevertheless, we will keep this combination of bribery and control in mind for future research.

We consider several special cases of multimode control, such as adding [16], deleting [16], or replacing [123] either candidates or voters. The following list gives an overview of the restrictions compared to the general multimode control problem:

We use a four-letter code for our problems. The first two characters CC/DC stand for construc-

Problem	Restrictions
Adding Voters	$\ell_{AC} = \ell_{DC} = \ell_{DV} = 0$ and $D = \emptyset$
Deleting Voters	$\ell_{AC} = \ell_{DC} = \ell_{AV} = 0$ and $D = W = \emptyset$
Adding Candidates	$\ell_{DC} = \ell_{AV} = \ell_{DV} = 0$ and $W = \emptyset$
Deleting Candidates	$\ell_{AC} = \ell_{AV} = \ell_{DV} = 0$ and $D = W = \emptyset$
Replacing Voters	$\lvert V' \rvert = \lvert W' \rvert$, $\ell_{AV} = \ell_{DV}$, $\ell_{AC} = \ell_{DC} = 0$, and $D = \emptyset$
Replacing Candidates	$\lvert C' \rvert = \lvert D' \rvert$, $\ell_{AC} = \ell_{DC}$, $\ell_{AV} = \ell_{DV} = 0$, and $W = \emptyset$

Table 2.3: Overview of several special cases of multimode control

tive/destructive control [10], the third character A/D/R means adding/deleting/replacing, and the last one C/V represents voters/candidates (for example, CCRV stands for constructive control by replacing voters). For simplicity, in each problem in the above table, we use ℓ to denote the integers in the input that are not necessarily required to be equal to zero. For example, when considering CCRV, we let ℓ denote the number of voters who may be replaced, that is, $\ell = \ell_{AV} = \ell_{DV}$. Similarly to bribery, we sometimes denote the final election by $(C, \bar{V})$ when studying control by adding and/or deleting voters or control by replacing voters. In contrast to bribery, the voters themselves in V and $\bar{V}$ may differ from each other, not only their rankings (as the chair deletes and/or adds some voters from/to the election). Note that we could use a similar notation for candidate control, but all our proofs get along without such a notation.

Sometimes, we place the letter "E" in front of the four-letter code, standing for the *exact* variant of control. For example, PLURALITY-ECCRV denotes the problem where the chair tries to make c a winner according to the Plurality rule by replacing *exactly* ℓ voters. Similar examples can be easily constructed.

In Chapter 1, several real-world applications of control by adding/deleting voters/candidates have been provided. In addition, multimode control models scenarios where an external agent tries several different attacks on a given election in order to reach his goal. Accordingly, a chair need not be content with only one attack (such as adding some voters), but might delete some voters and add some candidates as well, depending on his resources and influence. So multimode control appears to be a more flexible model than its special cases listed above. Nevertheless, both multimode control and "single" control attacks have their right to exist.

Replacement control models situations where the list of submitted ballots or the list of candidates must have a fixed size. Note that replacement control is no mere special case of multimode control as replacement control requires the additional restrictions [$\ell_{AV} = \ell_{DV}$ and $|V'| = |W'|$] (when voters are replaced) and [$\ell_{AC} = \ell_{DC}$ and $|C'| = |D'|$] (when candidates are exchanged).

We point out that there exist many more control types, such as control by partitioning voters or candidates [16], but these are not subject in this thesis (except for one partitioning problem, but only in Part II in a variant tailored to group identification). Nonetheless, these other control types

[10] We do not study destructive control or bribery in Part I. Nonetheless, we use the canonical four-letter code known from literature distinguishing between constructive and destructive problem variants.

are interesting for future research.

Let X be any type of control (we may argue analogously for bribery). A voting rule $\mathscr{F}$ is said to be *resistant* against X if the corresponding decision problem is NP-hard. For example, Plurality is known to be resistant against CCAC since PLURALITY-CCAC is NP-hard [16]. We further say that a voting rule $\mathscr{F}$ is *vulnerable* to X if $\mathscr{F}$-X is in P. A voting rule $\mathscr{F}$ is *immune* against X if the chair cannot make a candidate c a winner (when X is a constructive problem) or not a winner (when X is a destructive problem) unless c is already a winner of the election or not a winner of the election, respectively. For example, we know from [105] that Approval is immune to constructive control by adding candidates. Either c is already a winner with the highest approval score among all candidates or there is another candidate d with more points than c. This cannot by changed by the chair by adding arbitrary new candidates to the election since d remains in the election nonetheless and beats c in the final election, too. We further say that a voting rule $\mathscr{F}$ is *susceptible* to attack of type X if and only if $\mathscr{F}$ is not immune against X.

Chapter 3

Partial Information

As mentioned in Chapter 1, Konczak and Lang [117] generalized the canonical assumption with votes as complete rankings by allowing votes to be partial orders. With a slight abuse of notation, each voter's vote is given as a partial order when he specifies some pairwise comparisons for some pairs of distinct candidates (d,e). In other words, it is known whether a voter prefers d to e or the other way around. As a natural restriction, the voters' preferences satisfy the transitivity condition. Each partial order can be completed to a linear order without any contradiction [155].

The partial orders model is very flexible and captures several other, more concrete structures of partial information as we will see below in this chapter. It is no wonder that most research on partial information in voting has been dealing with partial orders since then.

One even more general partial information model was suggested in [39] wherein the authors regarded arbitrary partial profiles P (votes can be written in terms of any partial structure) and all complete profiles P' not contradicting P belong to the *information set* $I(P)$.

We adapt the notation with information sets the more so as this concept is known from game theory (cf. [96, 158]). We say that a profile $P' \in I(P)$ *extends* or *completes* P. Accordingly, we call P' an *extension* or *completion* of P. Likewise, we say that a vote v' extends or completes a partial vote v. We point out that, strictly speaking, an extension of a partial order may still be a partial order, but contains a little more information. In some applications, such as preference elicitation, the difference between "completion" (=an extension of a partial vote that is a complete ranking) and "extension" may play a crucial role. We do not make any difference between these two notions and use them interchangeably. Note that when we say "partial profile", this allows the profile to be complete as well, if not mentioned other. If we require a profile to be *properly partial*, we specify this aspect. A reason for this slight abuse of notation is that in case there is only one candidate, each vote can be regarded as partial and complete at the same time. Moreover, there are cases where votes are formally incomplete, but have only one possible completion and are thus complete in a sense. For example, we introduce a model according to which for each vote we know some positions and the candidates assigned to them. If there are two candidates in total and a voter assigns only position one to one of the two candidates, the vote is formally partial, but there exists merely one feasible completion of this vote.

As mentioned before, in case P is a complete profile (that is, each vote is given as a complete ranking), we have $I(P) = \{P\}$. In other words, the only way to feasibly extend a complete profile is

given by the profile itself. By contrast, if there is no information at all, $I(P)$ contains all thinkable profiles (cf. [39]).

Amongst others, the model in [39]—held very general—inspired us to have a closer look at the exact structure of partial information we are given in an election. Since under partial orders, many problems about winner determination (such as possible winner) or strategic behavior in elections might become hard, compared to the standard (and evidently unrealistic) model of full information, there may be yet further substructures with votes lying somewhere between partial orders and linear orders such that attacking the given election is yet an easy task for a malicious agent. Our goal is to determine the complexity both for partial orders and for other—to some extent—more concrete ways to display incomplete information.

Many such other structures of fractional information are thinkable or apply to some real-world situations. By way of example, suppose that each voter's four or five favorite alternatives are known, e.g., because of surveys, polls, or former elections. Assume further that Plurality is the voting rule used. Then a briber has enough knowledge and knows whom he has to bribe, in order to make a distinguished candidate a winner or prevent this candidate from winning, respectively.

The remainder of this chapter is organized as follows. First, Section 3.1 gives an overview of the most important literature concerning voting with partial information and strategic influences on elections. In Section 3.2, we introduce and recall nine different structures of partial information. Afterward, in Section 3.3, we present their interrelationships. Section 3.4 lays the focus on possible/necessary winner and solves some problems open up to now. In Section 3.5 and 3.6, we study the complexity of bribery under nine different partial information models. Finally, Section 3.7 concludes the chapter about partial votes.

3.1 Related Work

First of all, we have to mention the seminal papers by Bartholdi et al. [13, 14, 16] suggesting that NP-hardness of a voting problem (formulated as a decision problem) provides a certain protection against manipulative attacks—at least in the worst-case. These works encouraged many researchers to study the computational aspects of several problems related to voting.

Our works [30, 69] (as well as our works about necessary and possible voter control under incomplete information [143, 70] which are not subject in this thesis) fit into the line of research on the complexity analysis of winner determination, bribery, and control under some setting with incomplete information. Concerning voting under partial information, we point out the work by Konczak and Lang [117]. In their setting, votes are given as partial orders instead of linear orders. In their work, they introduced the possible/necessary winner problems. On the one hand, we complemented their work about the possible/necessary winner problems by establishing the complexity for several open problems concerning necessary/possible winner; on the other hand, we studied bribery and voter control in the style of the possible and necessary winner problems.

In the mean time, many other researchers have attended to the possible and necessary winner problem. Xia and Conitzer [167] studied these problems under partial orders for several voting rules, e.g., Plurality with Runoff, Maximin, k-Approval, and Bucklin. By contrast, Betzler and Dorn [21] laid their main focus on scoring rules and showed that, given partial orders, the possible winner problem is easy only for Plurality, Veto, constant scoring rules, and positive-affine transformations of these. Moreover, they revealed that possible winner is hard for almost all other scoring rules.

They left open the complexity for the scoring vector $(2,1,\ldots,1,0)$ which can be regarded as a hybridization of Plurality and Veto. Baumeister et al. [20] closed this gap and proved that possible winner for this scoring vector is hard given partial orders.

The possible winner problem was studied under different settings or restrictions to the partial orders model. Baumeister et al. [18] regarded a variant of possible winner where votes are top-truncated, bottom-truncated, or doubly-truncated orders. Note that these three models are considered in this thesis. Moreover, Chevaleyre et al. [32, 33] presented a setting where an election under complete information is given and new candidates join this election. Their model can be considered as a special case of our model 1TOS. They studied the possible winner problem for k-Approval, Borda, and some other scoring rules, and left open the complexity for the k-Veto family ($k > 1$). Additionally, other voting rules such as Bucklin and Maximin were considered under this setting [169]. The possible winner problem is further related to coalitional manipulation in a sense that possible winner given the CEV structure, defined in this chapter, is equivalent to coalitional manipulation. The coalitional manipulation problem is a generalization of single manipulation [14] and was introduced by [38] and further investigated in [36, 168, 170], to name a few. Remind that manipulation itself is not studied by us.

A probabilistic variant of possible winner was regarded in [7, 103]. The possible winner problem was further studied in [3, 12, 43, 44, 91] in various settings, more or less related to voting, but different from our setting.

Especially the work [39] inspired us as the authors did not commit themselves to the partial orders model, but regarded a general partial information model instead, admitting any kind of partial information. In their setting, a partial profile is given and the information set of this partial profile contains all complete profiles not contradicting the partial profile given. Although a problem related to voting can be hard under their general partial information model or given partial orders, we investigate whether the same problem is still hard under more concrete structures of partial information.

We have to mention some other papers dealing with some kind of uncertainty in voting. By way of example, in [19, 52, 64] a setting is given where there is uncertainty about the voting rule itself. The authors examined possible winner, manipulation, and bribery and control, respectively. Other works are about cp-nets [26, 50, 124], preference elicitation [42, 159], or incompleteness and incomparability in preferences [138] (the list is far from being complete). We further point out that Dey et al. studied manipulation under partial orders [46]. There is also a stream of research about voting rules directly applied to partial votes [59, 85, 127, 132] which differs from our setting.

Another line of research our works are based on is about strategic behavior in elections. The bribery problem was introduced in [78, 79]. The authors studied bribery for some voting rules such as Plurality, Veto, and Approval, and introduced some further concepts and generalizations of standard bribery, e.g., weighted, priced, or negative bribery. We further point out the work by Lin [119] who considered bribery in k-Approval and k-Veto. While both works are about bribery under complete information, we regarded bribery given incomplete votes. For bribery in other voting rules (under full information), we refer to the overview article "Control and Bribery in Voting" in [28] and the references therein. A dichotomy result for bribery in scoring rules can be found in [104]. The destructive bribery problem (or, more precisely, the related problem margin of victory) dates back to [165] where it was shown that destructive bribery is easy for all scoring rules. In contrast to Xia who considered the standard case with complete votes, we mainly consider constructive variants of bribery under incomplete information.

The bribery problem was studied in many variations and settings. In this context, we have to mention swap bribery [54] (where a briber cannot freely change a vote, but perform some pairwise swaps of adjacently ranked candidates in a voter's preference ranking), lobbying [25, 35], microbribery [80] (a bribery model tailored to intransitive preferences), bribery in party-based elections [174], or a protection model where an agent pays money to some voters to prevent them from being bribed [31]. To arbitrarily select some out off the multitude of works about bribery, we further refer to [17, 45, 50, 76, 99, 112, 144] for other works and settings more or less related to bribery,

Bartholdi et al. [16] initiated the research about many different types of electoral control. Amongst others, they studied control by adding voters and control by deleting voters. They considered and compared the voting rules Plurality and Condorcet. Hemaspaandra et al. [105] extended their research to destructive control. While they considered candidate control or control by partitioning voters as well, we merely regarded control by adding and deleting voters in our two works about control under partial information [143, 70] (which are widely irrelevant in this thesis, but fit in the scope of our work about strategic influences on elections under partial information).

Lin [119] studied control by adding voters, deleting voters, adding candidates, and deleting candidates for the voting rules k-Approval and k-Veto. Both Lin and Bartholdi et al., however, studied all problems under full information, whereas our focus lies on partial votes. Electoral control was investigated for many other voting rules. Once again, we refer to the overview article "Control and Bribery in Voting" in [28] and the references therein. Meanwhile, dichotomy results for scoring rules have been settled for constructive control by adding voters [107] and by deleting voters [104].

We point out that there are many other forms of control. By way of example, weighted control [82] and priced control [129] were studied. In these models, voters are equipped with weights and price-tags, respectively (similarly to bribery, cf. Section 2.3.3). Besides, there are some works about online control [108, 133]. A flexible model giving the chair the possibility to perform different control actions is given by multimode control [81]. Results about other control forms, such as control by equipartition of voters, runoff equipartition of candidates, multipartition of voters, or partition of voter groups, can be found in [65]. A recent paper generalizes destructive control in a way that a chair tries to prevent several candidates from winning an election [175]. Other forms of control exist, such as gerrymandering [109], voter deterrence [49], strategic candidacy [118], cloning [55], or control in the presence of manipulators [87]. Control was also investigated in different settings more or less related to voting, such as for multi-winner voting rules [141], in group identification [171], in judgment aggregation [17], or in fair division [6]. A recent model is about control through social influence [163]. All these settings are different from ours. Once more, we point out that the list makes no claim to be complete.

Our work also fits in the line of research characterizing voting rules by means of their complexity-theoretic behavior for different problems in voting. According to [128] and [63], the voting rules Normalized Range Voting and Fallback Voting are the best two voting rules known up to now in a sense that these two rules are vulnerable to only two out of 22 types of control. Up to now, to the best of our knowledge, no natural voting rule is known that yields only immunity or hardness results. We study in this chapter whether a setting with partial votes makes more problems hard or even impossible to the briber for the two classes of voting rules k-Approval and k-Veto.

Last but not least, we have to mention that—although in the main part of literature about winner determination and strategic influences in voting, the worst-case complexity is studied—several authors have recently investigated the parameterized complexity of possible winner, bribery, or con-

trol. See also [22, 24, 121, 161, 172, 173]. The list is not exhaustive.

3.2 Partial Information Models

In this section, we introduce nine models of partial information. Three models—Gaps, FP, and TOS—were introduced by us in [30] and further studied in [69, 70, 143]. The models PC, BTO, 1Gap (under the name *doubly-truncated orders*), TTO, and a variation of 1TOS had already been investigated in literature, whereas the CEV structure had been suggested in [117].

In the following, let (C,V) be an election with m candidates and n voters, and let the votes in V be incomplete according to the model X currently regarded. For each model presented in this section, the information set of a given partial vote or profile contains exactly the profiles not contradicting the given partial profile. Furthermore, we assume that all information provided by a model is the only information given.

3.2.1 Gaps

Our first partial information model handles the case where the briber only knows fractions of each ranking, i.e., there are some blocks in each vote that are completely ranked and there are some blocks for which it is known which candidates belong to them, but it is unknown how the candidates in such a block are ordered among themselves. By way of example, this model represents the cases where a voter is indifferent between alternatives or some alternatives are just incomparable to each other. In particular, the Gaps model can represent voters with weakly ordered preferences. We formally introduce this structure as follows (cf. [30]):

For each vote v, there is a partition $C_1^v,\dots,C_{2m+2}^v$ of the candidate set and a linear order (no information at all) for each C_j^v with j even (odd). Moreover, it holds $c^j \succ_v c^i$ for each $j,i \in \mathbb{N}$ $(1 \le j < i \le 2m+2)$ and all $c^j \in C_j^v$, $c^i \in C_i^v$. [1]

Note that possibly $C_j^v = \emptyset$ for some j. If $C_j^v = C_{j+1}^v = \emptyset$, we can drop both partite sets without changing the information set. [2] Therefore, we can restrict ourselves to at most $2m+2$ partite sets (an explanation will follow in Section 3.2.4).

Example 3.1.

(a) Complete votes v can be displayed by the Gaps model via $C_2^v = C$, $C_j^v = \emptyset$ $(1 \le j \le 2m+2$, $j \ne 2)$, and the given ranking for C_2^v is equivalent to the complete ranking over the candidates in C.

(b) Empty votes v yield $C_1^v = C$, $C_j^v = \emptyset$ $(2 \le j \le 2m+2)$ in the Gaps model.

The following example shows that the Gaps model allows some flexibility concerning the representation of a given vote:

[1] We point out here that, independently from our Gaps model, a somewhat similar notation for weak orders has occurred in [85]. In their setting, a weak preference order is written by $G_1^v \succ G_2^v \succ \dots \succ G_r^v$ for a given vote v, where in each G_i^v, $1 \le i \le r$, all candidates are tied. Our model is held more general as it allows several representations of the same vote, cf. the remainder of this section.

[2] For example, if a voter v yields the Gaps representation $C_1^v = C_2^v = C_j^v = \emptyset$, for each $j \in [2m+2]\setminus\{3\}$, and $C_3^v = \{a,b\}$, we may also display the vote by means of the sets $C_1^v = \{a,b\}$ and $C_j^v = \emptyset$ for all j, $2 \le j \le 2m+2$.

Example 3.2. *Let voter v have the preferences $a \succ_v b?c?d \succ_v e?f \succ_v g \succ_v h$ over the candidates in $C = \{a,b,c,d,e,f,g,h\}$. ($a?b$ means that the voter is indifferent between a and b.) There are different ways to write v according to the Gaps model:*

(a) $C_2^v = \{a\}$, $C_3^v = \{b,c,d\}$, $C_5^v = \{e,f\}$, $C_6^v = \{g,h\}$. *All other sets C_j^v ($1 \leq j \leq 2m+2 = 18$) are empty. We obtain the rankings $[a]$ and $[g \succ h]$ in the non-empty sets C_j^v with even indices.*

(b) $C_1^v = \{a\}$, $C_3^v = \{b,c,d\}$, $C_9^v = \{e,f\}$, $C_{10}^v = \{g\}$, $C_{13}^v = \{h\}$. *All other C_j^v ($1 \leq j \leq 18$) are empty. Voter v has the subranking $[g]$ according to the only non-empty partite set with even index (C_{10}^v).*

Two possible completions of vote v are $a \succ b \succ d \succ c \succ e \succ f \succ g \succ h$ and $a \succ c \succ b \succ d \succ f \succ e \succ g \succ h$.

The first version in (a) can be regarded as a minimum representation with as few partite sets as possible. This minimum representation guarantees that totally ordered partite sets are as large as possible. We could require such a minimum representation for each vote. [3] There are several reasons why we decided to define a more flexible version of Gaps.

One main reason is that we formally introduce the three following models as special cases of Gaps and this would not be possible in case we require a minimum representation (cf. the following sections).

There are further advantages of a flexible Gaps version. The second representation of voter v's ranking exploits that singleton sets (such as C_1^v and C_{13}^v) can be considered as candidate blocks representing full as well as no information at all: when there is only one candidate, on the one hand, we know all rankings concerning this candidate (because there is no ranking with other candidates); on the other hand, we do not know any pairwise comparisons and thus have no information. Our Gaps notation admits both interpretations.

The reason why g and h are further away from each other than in the first variant (where both belong to the same, totally ordered, partite set) may be that v perceives a larger distance between g and h than between the two candidates e and f on the one side and g on the other side. Thus, our flexible model could be used to display *cardinal preferences* (how large is the distance that v perceives between two candidates, see [139] as an example with cardinal preferences in voting) or *polarities* (e.g., v divides the candidates between very good, good, acceptable, and unacceptable candidates; even when v specifies a complete ranking, the distances between the candidate subgroups can be displayed by the Gaps structure). Polarities play a role, e.g., in sentiment analysis (cf. [98]). By way of example, as for the voting rule Approval, each voter has binary preferences. As each voter approves of some candidates and disapproves of all other candidates, the only given information is that each voter prefers an approved candidate to a disapproved candidate (this assumption has been suggested in various papers up to now). Formally, each voter approves of the candidates in C_1^v and disapproves of the candidates in C_3^v. Note that both partite sets are totally unordered. Also notice that the Approval rule can be directly applied under partial information according to this, i.e., we do not have to complete each voter's ranking and may immediately check which candidate is a winner.

[3] We presented an earlier version of Gaps at the workshops QBWL'14 and CASC'14. This original version required a minimum representation. However, since then, we have found some reasons why we decided to redefine Gaps by providing a more flexible model. Compare the following arguing.

We point out that the restriction to $2m+2$ has computational reasons. One can show that $2m+2$ partite sets suffice to exactly represent each possible ranking and to display some special substructures of Gaps (defined in the following sections). This minimizes the input size of an instance of Gaps. However, e.g., in order to display cardinal utilities, one may easily adjust the definiton of Gaps by defining the number of partite sets to be a sufficiently large constant $r \in \mathbb{N}$. For our subsequent complexity analysis, $2m+2$ partite sets suffice, for we are only interested in the voters' preference rankings.

3.2.2 One Gap

One special substructure of Gaps was introduced by Baumeister et al. [18] as *doubly-truncated preferences*, where in each vote there are subsets of candidates ranked at the top and at the bottom of the votes, and there is a gap between the top and bottom ranked candidates. 1Gap refers to the special case of Gaps with $C_j^v = \emptyset$ for each $j \in \{1\} \cup \{5, \ldots, 2m+2\}$, for each voter v. For the sake of simplicity, we assume that a voter v declares his top set C_T^v, middle set C_M^v (the gap), and bottom set C_B^v (i.e., we have $C_2^v =: C_T^v$, $C_3^v =: C_M^v$, and $C_4^v =: C_B^v$), plus complete rankings over C_T^v and C_B^v.

Votes submitted as doubly-truncated orders apply to situations where voters rank some of their most popular and some of their most unpopular candidates. In (political) elections, doubly-truncated orders make things easier for voters as they have the possibility to rank only few of their favorite and few of their most disliked candidates. Moreover, the 1Gap model applies to online rankings. For example, consider a film database where each voter ranks some of his favorite and some of his least favorite films, but due to the magnitude of films, the voter ranks only a tiny subsets of films.

Observe that 1Gap can display full information ($|C_M^v| \leq 1$ [4]), no information ($C_M^v = C$), and anything in between. For instance, regard a voter v with preferences $a \succ_v b \succ_v c?d?e?f \succ_v g$. In terms of 1Gap, we have $C_T^v = \{a,b\}$, $C_M^v = \{c,d,e,f\}$, and $C_B^v = \{g\}$, plus the rankings $[a \succ_v b]$ and $[g]$ for the sets C_T^v and C_B^v, respectively.

Note that we decided to keep the Gaps notion as well as the 1Gap structure flexible (by allowing singleton or empty partite sets not containing any information) in order to catch such situations with exactly one candidate in the middle set or with empty middle set and both top and bottom sets being non-empty. This will later turn out useful to reduce hardness results from 1Gap to the more general Gaps model. Similar "degenerate" cases exist for the next two models as well.

3.2.3 Top-truncated Orders

The model *Top-truncated Orders* (*TTO*, for short) was introduced by Baumeister et al. [18]. We equivalently define TTO as a special case of Gaps with $C_j^v = \emptyset$ for each $j \in \{1,4,\ldots,2m+2\}$ and each voter v.

Top-truncated votes apply to political or arbitrary other elections where each voter has to rank only an individual number of his favorite candidates, and all other candidates are assumed to be less

[4] Actually, full information refers only to the voter's ranking. Note that depending on the context, the cases $C_M^v = \emptyset$ and $|C_M^v| = 1$ can be interpreted differently. Moreover, the case $C_M^v = \emptyset$, $C_T^v \neq \emptyset \neq C_B^v$ is not equivalent to the case where $C_T^v = C$ or $C_B^v = C$. The latter two cases may model situations where a voter's complete ranking is known, but the voter is satisfied or dissatisfied with all candidates in C_T^v and C_B^v, respectively. Instances with precisely one candidate in the middle set can describe a voter unwilling or unable to utter his opinion about the middle set candidate.

preferred by the voter. The film example from the previous section can still be applied in a way that voters (=users) merely provide rankings over their top films instead of least favorite films.

For practical reasons, we assume that a given voter v specifies a top set $C_T^v := C_2^v$ and a bottom set $C_B^v := C_3^v$. Similarly to 1Gap and Gaps, top-truncated votes may be complete rankings (for $|C_T^v| \geq |C| - 1$). Again the word "complete" refers only to the voter's ranking, but not to the vote itself. Voters with one candidate in the bottom set can be interpreted differently (cf. the reasoning for the Gaps and 1Gap models).

Top-truncated votes can also be empty ($C_B^v = C$) or anything else. As an example, regard a voter v with the ranking $a \succ_v b \succ_v c \succ_v d?e?f$. In terms of the TTO model, we can rewrite this vote by $C_T^v = \{a,b,c\}$ and $C_B^v = \{d,e,f\}$, plus the ranking for C_T^v.

3.2.4 Bottom-truncated Orders

The structure *Bottom-truncated Orders* (*BTO*, for short) was also introduced by Baumeister et al. [18]. Similarly to TTO, we formally define BTO as a special case of Gaps with $C_j^v = \emptyset$ for each voter v and each $3 \leq j \leq 2m+2$. For convenience, we assume that a voter v specifies his top set $C_T^v := C_1^v$, his bottom set $C_B^v := C_2^v$, and a total order over the candidates in C_B^v.

Bottom-truncated votes apply to the context of online evaluations or sentiment analysis, where each user evaluates only negative contents of some items (alternatives) and their extent. In some of these applications, indifferences or ties between two evaluated alternatives are not allowed. All alternatives in a voter's top set can be regarded as immaculate and as preferred over all bottom set alternatives. Bottom-truncated votes further occur in some real-world voting processes or elections where each voter provides a ranking over his least favorite candidates. By way of example, consider a sports competition where some jury members—the voters in this context—vote against some of their least favorite candidates, and the candidate(s) with the largest number of vetoes are ruled out.

Analogously to the previous models, bottom-truncated orders can represent complete rankings ($|C_T^v| \leq 1$), empty votes ($C_T^v = C$), or anything between these two extremes. Suppose for instance that a voter v has the preferences $a?b?c \succ_v d \succ_v e$. These preferences can be expressed by the BTO model setting $C_T^v = \{a,b,c\}$ and $C_B^v = \{d,e\}$, plus the linear order $[d \succ e]$ in the bottom set.

Before we present the next partial information model, we give a short explanation why $2m+2$ partite sets suffice in the Gaps notation for any given number m of candidates. For the Gaps model, the most "wasteful" partition assigns exactly one candidate to $C_2^v, C_4^v, \ldots, C_{2m}^v$ each and all sets with odd index are empty. We therefore need no more than $2m$ sets as otherwise there are two adjacent empty sets and we can drop both of them and thus achieve yet a smaller partition (that is, the largest index j over all non-empty partite sets C_j^v becomes smaller, cf. the example in Section 3.2.1).

Our Gaps notation shall also be able to formally define the more special models 1Gap, TTO and BTO. We further require the first four partite sets $C_1^v, \ldots, C_4^v$ for any number of candidates as 1Gap requires the first four, TTO the first three, and BTO the first two partite sets. Thus, for any candidate number m, the smallest upper bound on the number of partite sets such that Gaps can capture all cases $m \in \mathbb{N}$, is $\max(4,3,2,2m)$. To obtain a compact expression valid for all natural numbers m, we fix $2m+2$ as the upper bound for all partite sets.

3.2.5 Complete or Empty Votes

There is a fourth substructure of Gaps. This model was suggested (but not further studied) by Konczak and Lang [117]. Formally, we may introduce CEV as a special case of TTO or BTO such that for each voter v the top set C_T^v or the bottom set C_B^v is empty. This structure is a generalization of the standard model of full information in a sense that new voters join the previous election. Absolutely nothing is known about these new voters, whereas we have (approximately) full (or enough [5]) information about the previous voters (due to former elections or surveys).

We assume that for a CEV instance the set of voters V can be divided into two partite sets V_c and V_e such that each vote in V_c is complete and each vote in V_e is empty.

Observe that this partition is unique for $|C| \geq 2$, whereas we can regard an arbitrary partition of V for $|C| = 1$ (for one candidate, we formally have full and no information at the same time). Note that, in contrast to the previous models, every vote is empty or complete, but nothing in between.

3.2.6 Pairwise Comparisons

PC, aka *partial orders*, is probably the most natural way to display partial preferences. For the first time, partial orders occurred in the context of voting in [117] and have been studied in many other papers since.

Formally, for each vote v we are given a subset $\Pi^v \subseteq C \times C$ such that the relation described by the pairs in Π^v is asymmetric and transitive.

By convention, $(a,b) \in \Pi^v$ means that voter v prefers a over b. As the title of this section suggests, we use the expressions *partial orders* and *PC* (or *pairwise comparisons*) interchangeably. Since the PC structure can also model intransitive preferences (in contrast to partial orders) by dropping the transitivity assumption, we decided to rename this canonical model. As intransitive preferences are not subject in this thesis, though raising intriguing questions for future research, we need not make any difference between these two notions.

As an example, let $C = \{a,b,c,d\}$ and assume that a voter v has the preferences $a \succ_v b$ and $c \succ_v d$. The information set of this vote contains the rankings $a \succ_v b \succ_v c \succ_v d$ as well as $a \succ_v c \succ_v d \succ_v b$, $c \succ_v a \succ_v d \succ_v b$, and three other rankings.

Note that the PC model can display no information ($\Pi^v = \emptyset$) or full information: suppose that voter v's preferences are $c^1 \succ_v c^2 \succ_v \ldots \succ_v c^m$, then an arbitrary $\Pi^v \supseteq \{(c^j, c^{j+1}) : j = 1, \ldots, m-1\}$, for which transitivity and asymmetry holds, represents v's preferences.

3.2.7 Fixed Positions

FP is another model introduced by us in [30]. Formally, for each vote v there is a subset of candidates C^v with distinct positions Pos^v in range between 1 and m assigned, i.e., there is a bijection $\beta^v : Pos^v \to C^v$ such that $\beta^v(p)$ is the candidate ranked to position p by voter v.

An example of this model is the case where there are three candidates c_1, c_2, and c_3. Candidates c_1 and c_3 have clearly opposing properties such that each voter prefers c_1 most and c_3 least or the other way round. Candidate c_2 is fixed to position 2, then. FP also captures situations where, by

[5] By way of example, when dealing with the Plurality rule, it suffices to know each voter's top candidate. It does not matter how the voters rank the candidates ranked on the positions $2, 3, \ldots, m$.

way of example, the first three positions and their candidates are known for each voter, but not the candidates behind them (note that such examples can be displayed by the 1Gap model, too).

Observe that FP can display full information ($|Pos^v| = |C^v| \geq |C| - 1$), no information ($Pos^v = C^v = \emptyset$), and anything in between. We point out once more that the information given under the FP model refers merely to the positions of the candidates in a given vote, but not to pairwise comparisons, polarities, or cardinal utilities.

As we will see later, the model has some interesting theoretical properties. So, FP is the only partial information model considered by us which is not a special case of the partial orders model. From the complexity-theoretic point of view, it is completely independent from PC as PC is not a special case of FP either. Verify that for the above example, with three candidates and c_2 fixed to position two, we do not know any pairwise comparison, but yet have some information—namely that c_2 is ranked second. As the information set contains only votes with c_2 ranked second, the information set of the given partial vote is a proper subset of the information set of an empty vote which obtains all potential rankings. Conversely, suppose that there are hundred candidates, a voter completely ranks 99 of them, and we do not know any pairwise comparison including the 100th candidate. Written in terms of FP, we have $Pos^v = \emptyset$ although we are far from dealing with a total lack of information.

3.2.8 Totally Ordered Subset of Candidates

In this section, we present yet another structure defined and introduced by us in [30]. In this model, for each voter we have information in form of a total order over an individual candidate subset. These subsets may differ from voter to voter.

Formally, for each vote v we have a subset C^v of candidates and a total order for C^v.

By way of example, such information can emerge in sentiment analysis (we refer the interested reader to [98] for an overview of sentiment analysis combined with voting) where a briber or a chair can extract information from each voter's previous comments or behavior (for example at giving scores for products bought on *eBay*). Or TOS may just reflect the case where we know for each voter an election this voter has participated in, and the voter's ranking over the candidates in this election. The example with the film database also applies to the TOS model when we assume that each voter (=user) specifies a ranking over an individual (tiny) subset of films and we do not know anything about the voter's preferences concerning the other films. These other films can be more or less preferred. There may be different reasons why the voter does not rank these other films. He could just forget to rank them, does not know them, or has no opinion about them. To even better capture such settings, we could generalize this model by admitting ties within the ranking.

Observe that we can display complete votes ($C^v = C$), empty votes ($|C^v| \leq 1$), or anything in between in terms of TOS. We point out that in both cases—$C^v = \emptyset$ and $C^v = \{d\}$ for some candidate $d \in C$—we do not know any pairwise comparison. In the latter case, it is yet known that v ranks the candidate d (but only compares d with himself); in the former case, v does not rank any candidates. Compared to analogous situations for other models, one could try to describe the differences between these two cases. One explanation could be that the "empty" voter has not participated yet in any election, whereas the "singleton" voter has already taken part in an election with just one candidate. Such differences, however, are immaterial to us since we merely study the computational complexity of a given problem.

Example 3.3. *Let (C,V) be an election with $C = \{a,b,c,d,e,f\}$, $V = \{v_1,v_2\}$, and a partial profile P such that v_1 votes $a \succ_{v_1} b \succ_{v_1} c$ (we have $C^{v_1} = \{a,b,c\}$) and v_2 has the ranking $c \succ_{v_2} e \succ_{v_2} f$ (and thus v_2's totally ordered subset is $C^{v_2} = \{c,e,f\}$).*

In Example 3.3, the partial rankings can be extended to the complete rankings ($a \succ_{v_1} d \succ_{v_1} e \succ_{v_1} b \succ_{v_1} c \succ_{v_1} f$ and $c \succ_{v_2} e \succ_{v_2} f \succ_{v_2} d \succ_{v_2} a \succ_{v_2} b$ as well as to many other completions not contradicting the two subrankings.

3.2.9 Unique Totally Ordered Subset of Candidates

A model similar to 1TOS was first suggested by Konczak and Lang [117] and further studied by Chevaleyre et al. [32, 33]. Chevaleyre et al. studied the possible winner problem for a variant of our 1TOS model with the restriction that the distinguished candidate must not be a new candidate in their setting. Hence, we could regard our model as a superstructure of theirs. Formally, 1TOS refers to the special case of TOS where $C^v = \tilde{C} \subseteq C$ for each voter v. A natural example here would be the addition of candidates to an election. An external agent might know the voters' preferences over the previous candidates, but has no information on how the voters would rank the new ones.

We point out that the 1TOS model generalizes full information and no information, but it is the only model regarded in this thesis for which either all votes are complete or empty or anything in between (provided that $|C| > 1$, otherwise each vote is empty and complete at the same time). In other words, given $|C| \geq 2$, an election (C,V) contains some complete (empty) votes if and only if all votes in V are complete (empty). Likewise, an election contains some neither empty nor complete votes if and only if all votes of the election are neither empty nor complete.

3.3 Hierarchy

In this section, we check which models are special cases of which other models, and which pairs of distinct models are independent from each other in a sense that we can find instances according to the first model but not the other model, and the other way round. Henceforth, we let

$$\text{PIM} := \{\text{PC}, \text{Gaps}, \text{1Gap}, \text{FP}, \text{TOS}, \text{BTO}, \text{1TOS}, \text{CEV}, \text{TTO}\}$$

be the set of all nine partial information models considered by us. Besides, we let $\overline{\text{PIM}} := \text{PIM} \cup \{\text{FI}\}$ where FI is the standard model of full information. Theorem 3.4 shows the relations between the partial information models discussed in this chapter.

Theorem 3.4. *The following relations hold:*

(1) 1TOS $\subsetneq$ TOS.
(2) CEV $\subsetneq$ TOS.
(3) CEV $\subsetneq$ TTO.
(4) CEV $\subsetneq$ BTO.
(5) TTO $\subsetneq$ 1Gap.
(6) BTO $\subsetneq$ 1Gap.
(7) 1Gap $\subsetneq$ Gaps.
(8) 1Gap $\subsetneq$ FP.
(9) TOS $\subsetneq$ PC.
(10) Gaps $\subsetneq$ PC.

This list is complete in the following sense: Relations that are not listed here and that do not follow from transitivity do not hold in general. The relationship of the partial information models is displayed in Figure 3.1 as a Hasse diagram.

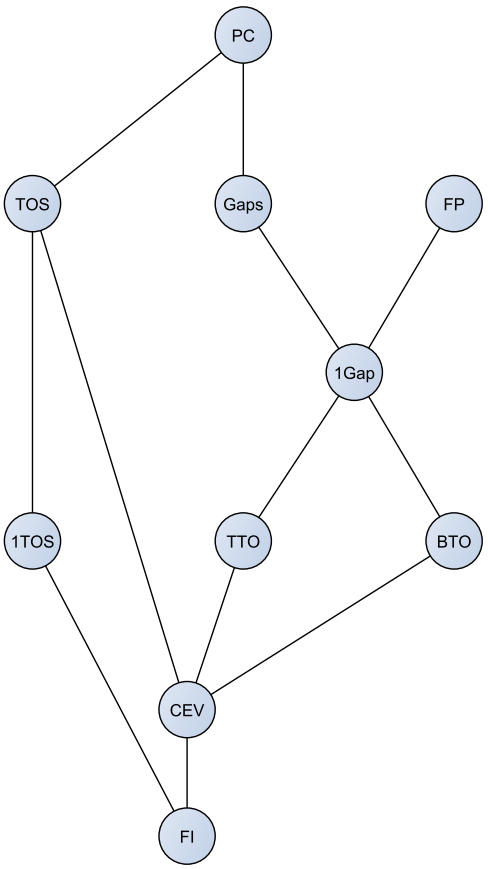

Figure 3.1: Hasse diagram

Proof. We first prove the ten relations, followed by points 11-15 showing incomparability for the remaining cases. Results following immediately from transitivity are not mentioned explicitly.

1. 1TOS $\subsetneq$ TOS. 1TOS$\subseteq$TOS holds due to the definition of the 1TOS model. We have 1TOS $\not\supseteq$ TOS because every voter ranks an individual subset of candidates under the TOS model in general.

2. CEV $\subsetneq$ TOS. Partial information according to CEV can be represented by partial information according to TOS with $C^v = C$ or $C^v = \emptyset$ for each voter v.

   The reverse direction is not true in general since for $|C| \geq 3$, $2 \leq |C^v| < |C|$, vote v cannot be represented in CEV. (Observe that for $|C| \leq 2$, the two models coincide as each vote is empty and complete at the same time (for $|C| = 1$) or either empty or complete (if $|C| = 2$) according to the TOS model.)

3. CEV $\subsetneq$ TTO. It holds CEV $\subseteq$ TTO since empty votes v may be written as $C_T^v = \emptyset$, $C_B^v = C$ in terms of the TTO model and we may set $C_T^v = C$, $C_B^v = \emptyset$ in the TTO model for complete

votes v. We have CEV $\not\supseteq$ TTO since voters v with both $C_T^v \neq \emptyset$ and $|C_B^v| \geq 2$ cannot be represented in CEV.

4. CEV $\subsetneq$ BTO. We have CEV $\subseteq$ BTO as follows. Empty (complete) votes v yield $C_T^v = C$ ($C_B^v = C$) in the BTO model. Moreover, it holds CEV $\not\supseteq$ BTO since $|C_T^v| \geq 2$, $C_B^v \neq \emptyset$ cannot be represented in CEV.

5. TTO $\subsetneq$ 1Gap. Any instance of TTO can be written as an instance of 1Gap with empty bottom set. We have TTO $\not\supseteq$ 1Gap since any 1Gap vote of the form $C_T^v \neq \emptyset \neq C_B^v$, $|C_M^v| \geq 2$ cannot be represented in TTO.

6. BTO $\subsetneq$ 1Gap. Any instance of BTO can be written as an instance of 1Gap with empty top set. The example proving TTO $\not\supseteq$ 1Gap also demonstrates that BTO $\not\supseteq$ 1Gap.

7. 1Gap $\subsetneq$ Gaps. Each instance of 1Gap is an instance of Gaps as 1Gap is defined as a special case of Gaps. We further have 1Gap $\not\supseteq$ Gaps since voters with $|C_1^v| \geq 2$, $|C_3^v| \geq 2$ written in terms of the Gaps structure (i.e., there are at least two "proper" gaps) cannot be represented in 1Gap.

8. 1Gap $\subsetneq$ FP. Let v be a vote according to the 1Gap model, i.e., v specifies a top set C_T^v, middle set C_M^v, and bottom set C_B^v, and $C = C_B^v \dot\cup C_T^v \dot\cup C_M^v$ holds. We obtain an instance of FP as follows. Set $C^v := C_T^v \cup C_B^v$ as the set of candidates assigned to a fixed position. Then $Pos^v := (\beta^v)^{-1}(C^v)$ contains all positions that voter v assigns to the candidates in C^v. In particular, $\beta^v(p)$ denotes the p-most preferred candidate in C_T^v ($1 \leq p \leq |C_T^v|$), while $\beta^v(|C_T^v| + |C_M^v| + p')$ represents the p'-most preferred bottom set candidate ($1 \leq p' \leq |C_B^v|$).

   The reverse direction is not true in general since partial information with only position 2 assigned to a candidate in a given vote cannot be represented in 1Gap if we have more than two candidates.

9. TOS $\subsetneq$ PC. Let $c^v(i)$ denote the i-most preferred candidate within C^v according to the partial information of type TOS. We can represent the information set by $\Pi^v = \{(c^v(i), c^v(j)) : 1 \leq i < j \leq |C^v|\}$ as partial information according to type PC.

   The reverse direction is not true in general since partial information $\Pi^v = \{(c_1, c_2), (c_3, c_4)\}$ cannot be represented in TOS when $c_1, \ldots, c_4$ are four pairwise different candidates.

10. Gaps $\subsetneq$ PC. According to the Gaps model, we denote the l-most preferred candidate within C_i^v by $c_i^v(l)$ for each even i (remind that for odd i, we do not know any pairwise comparisons within C_i^v). We can represent the information set by $\Pi^v = \Pi_1^v \cup \Pi_2^v$ where $\Pi_1^v = \left\{(c_i, c_j) : c_i \in C_i^v, c_j \in C_j^v, 1 \leq i < j \leq 2m+2\right\}$ and $\Pi_2^v = \{(c_i^v(l), c_i^v(l')) : 1 \leq l < l' \leq |C_i^v|, i \text{ even}\}$ as partial information according to type PC.

    The reverse direction is not true in general since voters v specifying the pairwise comparison $c_1 \succ_v c_2$ (that is, $\Pi^v = \{(c_1, c_2)\}$) cannot be represented in the Gaps model if we have more than two candidates: Let $c_3 \in C \setminus \{c_1, c_2\}$. Since c_3 can be ranked above, between, and below c_1 and c_2, we cannot find a unique block C_i^v containing c_3.

11. FP is incomparable to Gaps, PC, TOS, 1TOS. Let us start with FP $\not\subseteq$ Gaps. Consider a FP instance with five candidates c_j ($1 \leq j \leq 5$). Suppose that a voter assigns candidate c_3 to position 3 in the FP model. This example cannot be displayed by the Gaps model since we cannot decide for any other candidate $c_j \neq c_3$ whether c_3 is more or less preferred than c_j. Fixing c_3 and c_j in the same block with odd index in the Gaps model (i.e., without any information) is not possible either as then c_3's position is not fixed anymore.

    We further have Gaps $\not\subseteq$ FP since a Gaps instance with $C_1^v = \{c_1, c_2\}, C_2^v = \{c_3\}, C_3^v = \{c_4, c_5\}$ cannot be represented in FP. Only c_3's position is fixed in our Gaps instance. Where for Gaps the positions of c_1 and c_2 are among the first two places in v and c_4 and c_5 are ranked last and second to last by v, the FP model cannot ensure this anymore as all open positions can be arbitrarily assigned to candidates not fixed to any position.

    It futher holds FP $\not\subseteq$ PC since partial information with only position 2 assigned to a candidate cannot be represented in PC if we have more than two candidates. We further have PC $\not\subseteq$ FP. Let $C = \{c_1, c_2, c_3\}$ and $\Pi^v = \{(c_1, c_2)\}$. Then we do not know any fixed position and would obtain a FP instance with $Pos^v = C^v = \emptyset$. This in turn ignores the information $c_1 \succ c_2$ given in the partial orders model.

    (1)TOS $\not\subseteq$ FP since partial information with $2 \leq |C^v(\tilde{C})| < |C|$ and the associated order over C^v ($\tilde{C}$) cannot be represented in FP (since no position is fixed and v is neither empty nor complete). Conversely, we show (1)TOS $\not\supseteq$ FP as follows. Let $C = \{c_1, c_2, c_3\}$, $Pos^v = \{2\}$, and $C^v = \{c_2\}$ for a given vote v according to the FP structure. We do not know any pairwise comparison. Thus, C^v ($\tilde{C}$) has to be at most singleton in a (1)TOS instance. This, in turn, contradicts to our assumption that c_2 is fixed to rank 2 in vote v.

12. Gaps is incomparable to TOS, 1TOS. Gaps $\not\subseteq$ (1)TOS since Gaps instances with $|C_1^v| > 1$, $C_2^v \neq \emptyset$, and $C_l^v = \emptyset$ for each $l > 2$ cannot be represented in (1)TOS. (1)TOS $\not\subseteq$ Gaps since partial information $2 \leq |C^v|(|\tilde{C}|) < |C|$ and the associated order cannot be represented in Gaps.

13. (1)TOS is incomparable to 1Gap, BTO, TTO. (1)TOS $\not\subseteq$ 1Gap (BTO,TTO) since (1)TOS $\not\subseteq$ Gaps. Moreover, 1Gap $\not\subseteq$ (1)TOS, as 1Gap instances with $1 \leq |C_T^v|$ and $|C_M^v| \geq 2$ cannot be displayed by the (1)TOS model. Analogously, we show $X \not\subseteq$ (1)TOS for $X \in \{$BTO,TTO$\}$ via making use of the counterexample $2 \leq |C_T^v| \leq |C| - 2$. (The exemplary votes according to 1Gap, BTO, and TTO are neither empty nor complete, but at least one position is fixed in each case. Hence, the TOS model cannot represent this information given under the models 1Gap, BTO, and TTO, respectively.)

14. 1TOS is incomparable to CEV. 1TOS $\not\subseteq$ CEV as the case $2 \leq |\tilde{C}| < |C|$ (every voter ranks a subset neither complete nor empty) cannot be displayed by the CEV model. CEV $\not\subseteq$ 1TOS since elections containing both empty and complete votes (and $|C| > 1$) cannot be represented in the 1TOS model (since each voter has to rank the same subset of candidates under the 1TOS structure).

15. TTO and BTO are incomparable. TTO $\not\subseteq$ BTO since $2 \leq |C_T^v| \leq |C| - 2$ cannot be represented under the BTO structure. BTO $\not\subseteq$ TTO holds because of the same example.

Observe that 3.-6. also follow from [18]. We listed them here for the sake of completeness. The remaining relations follow from transitivity. ❑

Theorem 3.4 gives us a complete picture which models can be displayed by which other models, and which ones are incomparable to each other. This result will turn out useful when determining the complexity of a given problem. For example, since we know that possible winner for the Plurality rule is easy under the PC structure [21], the easiness result immediately follows for all substructures of PC. Since these substructures include all models in PIM $\setminus \{\text{PC}, \text{FP}\}$ and our problem is easy for PC, we achieve a P result for eight of nine models (all but FP).

Our next result reveals that and how for all votes v we can determine the candidates definitely, possible and not definitely, and not at all approved (vetoed) by v in polynomial time. For our purposes, we exploit Theorem 3.4 which shows as a byproduct that all special cases of FP or PC can be transformed in polynomial time to FP or PC, respectively.

Theorem 3.5. *Let $X \in$ PIM. Then determining the candidates definitely approved, possibly and not definitely approved, and definitely disapproved in each vote takes polynomial time. The same holds if we exchange "approved" with "vetoed".*

Proof. Let (C,V) be an election with a set C of m candidates, a set V of n voters, each of them being partial according to the same model $X \in$ PIM. W.l.o.g., we let $m,n \in \mathbb{N}$. We further let $m > k$ because otherwise all candidates are approved when studying the k-Approval rule. Thus, we even have $m > k \geq 1$.

It suffices to regard $X \in \{\text{PC}, \text{FP}\}$ as these two models are generalizations of every other model. By using Theorem 3.4, we can transform each other model into one of these two models. Observe that every transformation is polynomially bounded. We restrict ourselves to k-Approval as the arguing for k-Veto works nearly identically, by considering $(m-k)$-Approval instead of k-Veto.

Let us start with the FP model. Let Pos^v denote the set of fixed positions of voter v, C^v gathers all candidates assigned to fixed positions, and β^v is a bijective mapping $\beta^v : Pos^v \to C^v$ such that $\beta^v(p)$ is the candidate in C^v assigned to position $p \in Pos^v$ by voter v. First of all, we apply the following preprocessing.

First we check whether $[m] \setminus Pos^v \subseteq [k]$ or $[m] \setminus Pos^v \subseteq [m] \setminus [k]$. In other words, in such votes all open positions (if any) are either among the k approval positions or among the $m-k$ disapproval positions. Note that for a given vote v we can determine all candidates in $C \setminus C^v$ (that is, not assigned to any positions) in polynomial time. We simply fix all unassigned candidates arbitrarily to the open positions and obtain a complete vote according to this. As there are n votes in total, this preprocessing takes $O(m^2 n)$ time.

After performing this subroutine, each vote is either complete or has open positions both among the approval and disapproval positions, that is, $(C \setminus C^v) \cap [k] \neq \emptyset$ and $(C \setminus C^v) \cap ([m] \setminus [k]) \neq \emptyset$. For each vote, we can now partition the candidate set C into three partite sets:

- $C_A^v := \{d \in C : \ pos(d,v) \leq k\}$ (definitely approved candidates),
- $C_?^v := \{d \in C : \ d \notin C^v\}$ (possibly but not definitely approved candidates), and
- $C_D^v := \{d \in C : \ pos(d,v) > k\}$ (definitely disapproved candidates).

Observe that the subroutine and the determination of the partite sets C_i^v ($i \in \{A, D, ?\}$) take $O(m^2 \cdot n)$ time in total.

For PC, determining the three subsets C_i^v is a little more involved. W.l.o.g., we assume that for each vote v, the set Π^v of pairwise comparisons is maximal in a sense that it is non-extendable and contains all pairwise comparisons required to exactly describe voter v's ranking. In contrast, we can also represent Π^v in a compact manner. To illustrate what we have meant by this, suppose that a voter votes $a \succ b \succ c \succ d$. The economical representation $\Pi^v = \{(a,b),(b,c),(c,d)\}$ suffices to uniquely describe the voter's preferences since the remaining pairwise comparisons (a,c), (a,d), and (b,d) follow from transitivity. In our setting, we presume that $\Pi^v = \{(a,b),(a,c),(a,d),(b,c),(b,d),(c,d)\}$ represents voter v's preferences. Note that both notations may coincide. For example, let voter v prefer a over b and c over d, and this is all information given. Then $\Pi^v = \{(a,b),(c,d)\}$ is the unique way to describe voter v's preferences.

To determine the three partite sets C_A^v, $C_?^v$, and C_D^v for each vote v, the following finding helps us:

- $d \in C$ is definitely approved of by v if there are at least $m-k$ candidates e such that $d \succ_v e$. [6]
- d is definitely disapproved by v if there are k or more candidates e with $e \succ_v d$.
- d is possibly but not definitely approved by v if v prefers d over at most $m-k-1$ candidates and there are at most $k-1$ candidates that voter v prefers to d (in other words, neither of the previous two cases hold).

Notice that this subroutine takes $O(m^3 n)$ as there are n voters, at most $\frac{m(m-1)}{2}$ pairwise comparisons to check for each vote, and we count the number of won and lost pairwise comparisons for each candidate in a given vote. ❑

Henceforth, we will follow the common practice in COMSOC literature and assume that Theorem 3.5 holds, especially when showing membership in P. In other words, we will implicitly assume that for any model in PIM, for each vote we can determine in polynomial time the candidates definitely approved, possibly and not definitely approved, and definitely disapproved. The same holds when we replace "approvals" by "vetoes". Note that we often transform instances of other models such as Gaps or TOS into equivalent instances of PC or FP for which we additionally use Theorem 3.4. In other words, the studied model itself is in $\{\text{FP}, \text{PC}\}$ or can be transformed in polynomial time into one of these two models. In this context, Theorem 3.4 does not only show that such a transformation is in P, but also how to construct such a transformation. Note that the complexity for a problem under a model $X \in \text{PIM} \setminus \{\text{FP}, \text{PC}\}$ may be slightly easier than for FP or PC, by directly exploiting the special structure of X instead of transforming the information to an equivalent instance under FP or PC. Since we only wish to check whether or not a problem can be decided in polynomial time, we do not care about the best possible complexity of a problem as long as the problem turns out to be easy.

3.4 Necessary and Possible Winner

In this section, we examine two of the most basic problems dealing with partial information in voting—*Possible Winner* and *Necessary Winner*—which were introduced by Konczak and

[6] To count the number of these candidates, we need the maximal representation of Π^v.

Lang [117]. In their pioneering work, votes are given as partial orders instead of linear orders which had been the canonical assumption up to then. While votes given as strict linear orders rank all candidates from first to last without any ties, votes given as partial orders reveal only some partial information in terms of some known pairwise comparisons between pairs of distinct candidates. The possible and necessary winner problems are defined as follows (cf. [117]), we adapt this definition to any arbitrary partial information model.

$\mathscr{F}$-X-POSSIBLE/NECESSARY WINNER	
Given:	An election (C,V) with a set C of m candidates, a set V of n voters according to $X \in \overline{\text{PIM}}$, and a designated candidate $c \in C$.
Question:	Is c a winner of the election under $\mathscr{F}$ for at least one/for every completion of the votes in V?

Both problems can be regarded as two sides of the same coin. The necessary winner problem requires a candidate to win regardless of how we extend the partial profile to a complete profile, while the possible winner problem requests that this candidate wins for at least one extension. One could also speak of the pessimistic or optimistic approach, or of the "for all" and "there exists" variant. Notice that necessary winners are possible winners, but the opposite direction does not hold in general.

The necessary winner problem comprehends some kind of robustness property. Since some kind of uncertainty occurs in most real-world settings, the question arises whether a candidate c wins a given election, independently of the way we resolve the existing incompleteness.

Possible winner can be interpreted as a best-case scenario or, equivalently, a minimum criterion that a given candidate must satisfy. Moreover (cf. [146]), there is another interpretation of the possible winner problem: voters may have partial rankings because of some kind of indecisiveness or indifference towards some alternatives. Accordingly, possible winner asks whether there is a way to convince the voters to decide for one complete vote, not contradicting to their partial rankings.

In this remainder of this chapter, we will consider possible/necessary winner under all nine models in PIM and solve almost all open problems. Table 3.1 gives an overview of the state-of-the-art:

Voting rule	PC	CEV	1TOS	1Gap	TTO	BTO
Plurality	P [21]	P	P	P	P	P
2-Approval	NPC [167]	P [36]	P [33]	P [18]	P [18]	P [18]
k-Approval ($k \geq 3$)	NPC [167]	P	NPC [33]	P [18]	P [18]	P [18]
Veto	P [21]	P	P	P	P	P
k-Veto ($k \geq 2$)	NPC [21]	P [36]				

Table 3.1: Complexity results for the possible winner problem under the co-winner model known so far. Key: P stands for "polynomial-time solvable", whereas NPC means "NP-complete". For some polynomial-time results following from more general results, we merely indicate the sources of the most general polynomial-time results. So, e.g., PLURALITY-CEV-POSSIBLE WINNER is known to be easy from [14], but the complexity of this problem also follows from the more general problem PLURALITY-PC-POSSIBLE WINNER.

Observe that Table 3.1 includes only six of nine models in PIM. Furthermore, the table itself contains several open problems.

In opposition to possible winner, we know that the necessary winner problem is easy for every scoring rule under partial orders. As all models in PIM $\setminus$ {FP,PC} are special cases of PC, they all inherit the polynomial-time upper bound. Hence, necessary winner is open only under the FP structure.

In this section, we solve almost all these open problems under the co-winner model and contribute the following results:

- We close the gaps for the FP model, regarding necessary and possible winner. Both possible and necessary winner are easy for both families of voting rules given FP. Moreover, we find that the P result for necessary winner under the FP model can even be extended to all scoring rules.
- For 1TOS, we provide a dichotomy for the possible winner problem for the k-Veto family. In particular, we show that the problem is easy for $k \leq 3$ and hard for $k \geq 4$ under the co-winner model. In addition, we show that 3-VETO-1TOS-POSSIBLE WINNER is hard under the unique winner model. According to this, ties matter for 3-Veto.
- We settle the complexity for some open problems under the TOS model.
- We close the gaps in Table 3.1 by pointing out that possible winner in k-Veto is easy under the models 1GAP, BTO, and TTO.

All these new results have been presented by us at my talk held at the Doctoral Consortium in October 2017 (within the scope of the conference *ADT 2017* in Luxembourg).

For the remainder of this section, if not mentioned other, our input is always an election (C,V) with m candidates in candidate set $C = \{c_1, \ldots, c_{m-1}, c\}$, n voters in voter set $V = \{v_1, \ldots, v_n\}$ (where $m \in \mathbb{N}$ and $n \in \mathbb{N}_0$), and a designated candidate $c \in C$. Moreover, all votes are partial according to the same model $X \in$ PIM. We assume that $m > k$ when regarding k-Approval or k-Veto. Otherwise, the problem would be trivial. If $m = 1$, c—the only candidate—is always a possible and necessary winner under both winner models. For $m > 1$ and $m \leq k$, c is a possible and necessary winner under the co-winner model, whereas c is neither possible nor necessary winner under the unique-winner model. We never consider such trivial cases. At this place, we must also point out that in literature k-Approval/Veto is sometimes not even defined for $m < k$. Unless stated otherwise, all proofs and results are presented under the co-winner model.

At times, we write c_j, small letters a, d, e, or similar symbols to denote general candidates or general non-distinguished candidates or to adapt our notation to the special structure of a partial information model, such as 1TOS with two partite candidate sets $\tilde{C}$ and $C \setminus \tilde{C}$. Throughout this chapter, we further let $score(d)$ and $vetoes(d)$ denote the (definite) score and veto number of a candidate $d \in C$, respectively. These numbers refer to the case where the set of voters is clear from the context. The more precise expressions $score_{(C,V)}(d)$ and $vetoes_{(C,V)}(d)$ indicate the score and veto number of d in election (C,V), respectively. Moreover, $score_{(\tilde{C},V)}(d)$ provides us the score of candidate d restricted to subelection $(\tilde{C},V)$. Likewise, we can define similar expressions for the veto number or for larger or smaller sets of voters. $pscore(d)$ denotes the maximum number of points candidate d can reach for any (best-case) extensions of a given partial profile, $pvetoes(d)$ in turn counts all definite and uncertain vetoes for candidate d. We can similarly define expressions such as $pvetoes_{(C,V')}(d)$ or $pscore_{(\tilde{C},V)}(d)$ (with $V' \subseteq V$, $\tilde{C} \subseteq C$). The expression $score(a,b)$ denotes the

number of voters (definitely) approving of both candidates a and b. Likewise, we define $\mathit{vetoes}(a,b)$. Besides, we sometimes write expressions such as $\mathit{vetoes}(c,\neg c_j)$ or $\mathit{score}(\neg c_i,\neg c_j)$. The former expression denotes the number of voters vetoing c and not vetoing c_j, while the latter expression returns the number of voters neither approving of c_i nor c_j. Once again, we can construct similar expressions, such as $\mathit{score}_{(C,V)}(\neg c, c_j)$.

We say that a voter v *definitely* approves of a candidate d in a partial profile P if and only if v approves of d under all extensions of P. In a similar manner, v *potentially* approves of d if and only if v approves of d for at least one complete profile in $I(P)$ if and only if v either definitely approves of d or possibly but not definitely approves of d. Last but not least, v (definitely) disapproves of d in a partial profile P if and only if v does not potentially approve of d. We obtain analogous notions by replacing "approve of" by "veto".

Voting rule	FI	Gaps	FP	TOS	PC	CEV	1TOS	1Gap	TTO	BTO
Plurality	P	*P*	*P*	*P*	P [21]	P	P	P	P	P
2-Approval	P	*P*	*P*		NPC [167]	P	P [33]	P [18]	P [18]	P [18]
k-Approval ($k \geq 3$)	P	*P*	*P*	*NPC*	NPC [167]	P	NPC [33]	P[18]	P[18]	P [18]
Veto	P	*P*	*P*	*P*	P [21]	P	P	P	P	P
2-Veto	P	*P*	*P*		NPC [21]	P	**P**	*P*	*P*	*P*
3-Veto	P	*P*	*P*	**NPC**	NPC [21]	P	**P**	*P*	*P*	*P*
k-Veto ($k \geq 4$)	P	*P*	*P*	**NPC**	NPC [21]	P	**NPC**	*P*	*P*	*P*

Table 3.2: Complexity results for the possible winner problem under the co-winner model. Results in boldface are new. We point out that the results for k-VETO-X-POSSIBLE WINNER, $k \in \mathbb{N}$, $X \in \{\text{BTO,TTO,1Gap}\}$, immediately follow from the corresponding algorithm for k-Approval in [18], albeit not pointed out in that work. That's why the results are written in italic. The same holds for Gaps and FP under both the k-Approval and k-Veto families. To show these results, one may apply the algorithm in [18], too. Generally, results written in italic, albeit new, follow from known existing results via reductions (e.g., PLURALITY-TOS-POSSIBLE WINNER). All other results are known from literature.

Now we are equipped to provide the first complexity results. Let us start with necessary winner which is open merely for the FP model. Remind that necessary winner asks whether a candidate c is a winner for every way to complete a partial profile to a complete profile. We do not only provide a result for k-Approval and -Veto but for all scoring rules.

Theorem 3.6. *$\mathscr{F}$-FP-NECESSARY WINNER is in* P *for every scoring rule $\mathscr{F}$.*

Proof. Our algorithm breaks our instance down into $m-1$ subinstances I_j ($1 \leq j \leq m-1$), one for each c_j, and checks for each subinstance whether c_j beats c for an extension where c_j ranks on the best possible position and c performs as badly as possible in every vote. Observe that those completions maximize the score difference between c_j and c. If c ties with or beats c_j in such extreme extensions, c ties with or beats c_j in all other extensions. It follows from the arguing up to now that our instance of necessary winner is a YES instance if and only if each subinstance I_j is a NO instance, that is, for each c_j, the worst-case score of c is still at least as high as the best-case score of c_j.

For subinstance I_j, we apply the following subroutine for each vote v_i ($i \in [n]$):

- If v_i does not assign any fixed position to c_j, that is, $c_j \notin C^{v_i}$, we fix c_j on the best possible

open position $\min\{p : p \in [m] \setminus Pos^{v_i}\}$. If $c \notin C^{v_i}$, reset c on the worst possible open position $\max\{p : p \in [m] \setminus Pos^{v_i}\}$.

After performing this algorithm for every vote, we obtain unique positions both for c_j and c and it remains to check whether $\sum_{i=1}^{n}(score_{(C,\{v_i\})}(c_j) - score_{(C,\{v_i\})}(c)) > 0$ holds. Then (and only then) c_j beats c in an extension where c_j is ranked best possible by each voter and c ranks on the worst possible position in each vote (only relevant when the positions of c_j and c are open in a given vote, respectively).

Our algorithm regards at most $m-1$ non-distinguished candidates and checks whether one of them beats c for any extension. For every pivotal candidate, the algorithm determines for every vote which positions are open, if c and/or c_j are assigned to any fixed position, and adjusts their positions if necessary (in case one or both of them is/are not assigned to any position). Finally, the total score difference is calculated. Since there are n votes and each vote has m positions, our procedure takes $O(mn)$ time for each pivotal candidate, that is, subinstance I_j. As there are $O(m)$ pivotal candidates to check, our entire algorithm takes $O(m^2n)$ time. ❑

When facing a problem related to necessary winner (such as necessary bribery), we basically try to make the worst-case of c at least as good as the best-cases of all other candidates. (This arguing meets limits as we will see next and in several later proofs.) Referring to necessary winner under the FP model, we may do so as there are extreme extensions of our partial profile where c is ranked on the worst possible open position in each vote not assigning a fixed position to c, and each other candidate c_j performs as well as possible in a sense that a voter either assigns a unique position to c_j or, in case c_j is not assigned to any position, we assume that c_j is ranked on the best possible open position by the given voter. This can be done for all non-distinguished candidates at the same time. For example, when a voter possibly but not definitely ranks the non-distinguished candidates a, b, and d first, given a partial profile, and our underlying voting rule is Plurality, we may assume that all three candidates simultaneously get the point as for each candidate e in $\{a,b,d\}$ there are extensions of the partial profile where e receives all potential points.

Observe that, under the FP model, it is a necessary and sufficient condition for c to be a necessary winner that the worst-case of c is at least as good as the best-cases of all non-distinguished candidates. This condition is necessary since such extreme extensions may actually occur for every c_j. It is sufficient as well because in case c's worst-case is at least as good as all other candidates' best-cases, c is strong enough in all other extensions since all other completions are not worse for c. In other completions, c has at least as many points and/or some c_j has no more points. This arguing also takes into account that each scoring rule is *weakly monotonic*. (Weak monotonicity of a voting rule is defined as follows [146]. Suppose that c is a winner of a given election. Then c remains a winner if a voter raises c higher in his ranking and leaves the rankings according to the other candidates unchanged, e.g., when the voter initially has the ranking $a \succ b \succ c \succ d \succ e$ and changes his preferences into $c \succ a \succ b \succ d \succ e$.) For Gaps and all its substructures, the same condition is equivalent to c being a necessary winner.

As we could observe for partial orders and as we will see for (1)TOS in some proofs, this holds only to some extent for these models. Nonetheless, necessary winner is easy for all three models for all scoring rules, due to [167] and the fact that 1TOS and TOS are special cases of PC.

We point out that the proof for necessary winner under the PC structure can be found in [167] and is a little more involved. If a voter prefers c to a non-distinguished candidate c_j (that is the

pivotal candidate), we cannot rank c as bad as possible as this worst-case for c may be actually good for c. Consider the following example:

Example 3.7. *Let (C,V) be an election with $C = \{c, c_1, \ldots, c_4\}$, $V = \{v_1, \ldots, v_5\}$, distinguished candidate c, and let $\mathscr{F} =$ Veto. The partial votes are defined as follows, according to the PC model:*

- *The voters $v_1, \ldots, v_4$ are complete and veto c, c_2, c_3, and c_4, respectively.*
- *Voter v_5 prefers c over c_1. No more information is given for voter v_5.*

Observe that all candidates but c_1 are definitely vetoed by exactly one voter each. Moreover, v_5 possibly but not definitely vetoes all non-distinguished candidates. When fixing c as bad as possible in v_5, this voter ranks c on position four and c_1 on position five (in this case, c loses all pairwise comparisons in doubt). In c's "worst-case", c is a winner because all candidates are vetoed exactly once. In all "better" cases where c and c_1 are ranked better than on position four and five by voter v_5, respectively, c_1 is the only winner with zero vetoes. This example describes a paradox for partial orders in a sense. One explanation for this paradox is that v_5 must rank c on a better position than c_1 although no position in v_5 is fixed. According to this, the proof in [167] tailored to partial orders checks where to rank the candidates c and c_j (and potentially some alternatives definitely lying in between) such that the score difference $score_{(C,\{v\})}(c) - score_{(C,\{v\})}(c_j)$ is as small as possible for a given vote v. In our example, this difference between c and c_1 in v_5 is zero, fixing c on position four leads to a difference of one.

By contrast, for the FP model we can simply assign c to the worst possible position and c_j to the best possible position in each vote (in case c's or c_j's position is open, respectively)—regardless of the other candidate's position.

Note that, not least because of Theorem 3.6, we can decide in polynomial time for all nine models and all scoring rules $\mathscr{F}$ whether a given candidate is a necessary winner or not.

Unfortunately, we cannot argue for possible winner that the best-case for c beats or ties with all worst-cases of the c_j. This condition is surely necessary for c to be a possible winner, but far from being sufficient as the worst case of some c_j often excludes the worst case of some other non-distinguished candidate and vice versa. Given partial orders, regard the following example of Plurality with $C = \{a,b,c\}$ and one voter voting $a \succ c$ and $b \succ c$. Either a or b is ranked first. c gets zero points in his best-case and both a and b have no single point in their worst-cases. However, the worst-case of a excludes the worst-case of b and the other way around. To determine if c is a possible winner or not hence requires some more involved approaches and due to [21] we know that possible winner under partial orders is only in P for the most basic scoring rules such as the constant rule, Plurality, and Veto (and all positive-affine transformations) and NP-complete for each other scoring rule [21, 20].

In the following, we restrict ourselves to two important subclasses of scoring rules—k-Approval and k-Veto—for which we want to provide dichotomy results, i.e., the values k for which possible winner is easy and for which this problem is hard. Our next result manifests that for six models—Gaps, FP, 1Gap, BTO, TTO, and CEV—possible winner is easy for k-Approval and k-Veto for each value of k. Especially for Gaps and FP, very general models for themselves (cf. the Hasse diagram in Figure 3.1), this result appears to be rather astonishing at first sight as under partial orders the possible winner problem is hard even for 2-Approval [167] and 2-Veto [21].

Theorem 3.8. *k-APPROVAL-X-POSSIBLE WINNER and k-VETO-X-POSSIBLE WINNER are in* P *for each $k \in \mathbb{N}$ and each model $X \in \{Gaps, FP, 1Gap, BTO, TTO, CEV\}$.*

Proof. It suffices to argue only for Gaps and FP. Showing membership in P for these two models, the easiness result immediately follows for all four other models due to Theorem 3.4. Let $X \in \{\text{FP}, \text{Gaps}\}$ be an arbitrary but fixed model. Let us begin with k-Approval. We can directly exploit the flow algorithm by Baumeister et al. [18] tailored to doubly-truncated orders (aka 1Gap) as, both for Gaps and FP, we only have to distinguish between definite approvals, unsure approvals, and definite disapprovals in each vote v. Moreover, in case there are k_1 definite approval positions and k_2 unsure approval positions in a given vote, and the voter possibly but not definitely approves of k' candidates (observe that it holds $k_2 < k'$), the voter can approve of arbitrary k_2-combinations over these k' candidates for some extensions. For instance, when $k_2 = 2$, $k' = 4$, and the candidates $c_1, \ldots, c_4$ are possibly but not definitely approved in a given vote, each 2-subset of $\{c_1, \ldots, c_4\}$ can be approved by this voter for some extensions. We will see below that this does not hold for all models in PIM.

According to this, we obtain the same setting as under the 1Gap model for which the proof was shown in [18]. The proof by Baumeister et al. [18] can be directly tailored to k-Veto by considering $(m-k)$-Approval instead.

In the Appendix, we will provide a proof sketch with bipartite matching (instead of flow maximization), strongly based on the proof idea in [18]. ❑

Now let us turn to the three remaining models—PC, TOS, and 1TOS. We cannot adapt the proof of Theorem 3.8 to any of these models as they inhere additional structure in general. What is meant by this, is explained by means of the following example:

Example 3.9.

1. *Let $C = \{a, \ldots, d\}$, $\mathscr{F}$ =2-Approval, $X \in \{1TOS, TOS\}$, and suppose that a voter v ranks the candidates in $\tilde{C} := \{a, b\}$ by $a \succ_v b$. (We w.l.o.g. argue for the 1TOS structure in the first and second example, the reasoning under the TOS model works analogously.)*

2. *Let $C = \{a, \ldots, e\}$, $\mathscr{F}$ =3-Approval, $X \in \{1TOS, TOS\}$, and assume that a voter v ranks the candidates in $\tilde{C} := \{a, b, c\}$ via $a \succ_v b \succ_v c$.*

3. *Let $C = \{a, \ldots, g\}$, $\mathscr{F}$ =4-Approval, X =PC, and a voter v specifies the pairwise comparisons $\Pi^v = \{(a,b), (a,c), (a,d), (b,f), (c,e), (c,g), (d,g)\}$.*

In the first example, all candidates are possibly but not definitely approved by v. Note that in case v approves of b for a given extension, v approves of a, too. The following four pairs of candidates can be approved for some extensions: (a,b), (a,c), (a,d), and (c,d); the other two combinations, (b,c) and (b,d), would hurt the condition $a \succ b$.

In the second example, the following triples of approved candidates may occur for some completions: (a,b,c), $(a,b,*)$ $(* \in \{d,e\})$, and (a,d,e). Candidate a is definitely approved, all other candidates are possibly but not definitely approved by v. For a given extension, an approval for c implies approvals for both a and b, and once again, an approval for b implicates an approval for a. Such interdependencies and the inherent additional structure intuitively make the possible winner problem harder for the models 1TOS and TOS. Nevertheless, keeping the first and second example

in mind, these dependencies move only in "one and the same direction" under the TOS and 1TOS structures.

For PC, the structure may be much more complex as the third example suggests. Verify that a is preferred to all other candidates and is hence definitely approved (as a wins all pairwise comparisons). All other candidates are possibly but not definitely approved by v. Table A.1 in the Appendix will give an overview of all feasible 4-combinations of approved candidates for this example.

We point out that for Plurality and Veto such interdependencies do not occur either. If $a \succ_v b$ holds in a vote v, this excludes b from being approved and a from being vetoed. It is hence no surprise that these two voting rules are the only non-trivial scoring rules for which possible winner is easy under partial orders [21]. For 2-Approval (or higher values of k), b can yet be approved by voter v although it is known that $a \succ_v b$.

As possible winner under partial orders has been studied even for all scoring rules and in particular for k-Approval/-Veto, it remains to have a closer look at the models 1TOS and TOS. The following results reveal that under the 1TOS structure there are additional polynomial-time results for possible winner in k-Approval and k-Veto, but nevertheless almost all problems are hard. From [33], we know that the possible winner problem is in P for Plurality and Veto (also follows from [21] as PC is a generalization of 1TOS) and 2-Approval and NP-complete for k-Approval ($k \geq 3$). For k-Veto, we do not know the complexity of k-VETO-1TOS-POSSIBLE WINNER for $k \geq 2$. We provide the missing results next. Let us start with $k = 2$.

Theorem 3.10. 2-VETO-1TOS-POSSIBLE WINNER *is in* P.

We omit the proof at this point because we will later show that the more general problem possible bribery is in P (cf. Theorem 3.41 in Section 3.6). This result implicates membership in P for 2-VETO-1TOS-POSSIBLE WINNER.

Theorem 3.11. 3-VETO-1TOS-POSSIBLE WINNER *is in* P *under the co-winner model.*

Proof. Let $\tilde{C} \subseteq C$ denote the subset completely ranked by every voter in V. W.l.o.g., we assume that $|C| > 5$. The case $|C| \leq 3$ cannot occur (as we have premised that k-Veto requires at least $k+1$ candidates in C), for $|C| \in \{4,5\}$ our voting rule is either Plurality or 2-Approval for which the results can be found in [21] (for Plurality since 1TOS is a special case of PC) and [33] (for 2-Approval). To show membership in P, our algorithm distinguishes the following cases:

- $c \in C \setminus \tilde{C}$ or $|\tilde{C}| \leq |C| - 3$. In either case, there are extensions for which c is not vetoed by any voter. We accept since no other candidate can achieve fewer vetoes than c for these extensions.

- $C = \tilde{C}$. In this case, each vote is complete. As possible winner and the winner problem coincide under full information, membership in P follows from [15].

- $c \in \tilde{C}, C \setminus \tilde{C} = \{p\}$. In this case, each voter definitely vetoes his two least preferred candidates in $\tilde{C}$, and possibly but not definitely vetoes p and his third to least preferred candidate in $\tilde{C}$. If c is a voter's third to least preferred candidate in $\tilde{C}$, we assume that the voter does not veto c, but certainly vetoes his two least preferred $\tilde{C}$ candidates and p. With this setting, we may apply the flow algorithm defined by Baumeister et al. in the context of doubly-, bottom-, and top-truncated votes (cf. the proof of Theorem 3.8 and [18]).

- $c \in \tilde{C}$, $C \setminus \tilde{C} = \{p,q\}$. Now there are two candidates in $C \setminus \tilde{C}$. To show membership in P for this subcase, there are two possibilities. On the one hand, one can adapt the proof in [33] (more precisely, Lemma 1 and Proposition 5) showing that k-APPROVAL-1TOS-POSSIBLE WINNER is in P for each $k \in \mathbb{N}$ when there are two new candidates different from c, that is, $c \in \tilde{C}$, $C \setminus \tilde{C} = \{p,q\}$, and $p \neq q$ holds.

  A second proof, directly tailored to this subcase of 3-VETO-1TOS-POSSIBLE WINNER, will be given in the Appendix. In contrast to the proof in [33] specific to k-Approval, we will provide a completely different approach arguing by means of generalized b-edge cover. The argument of correctness follows similarly to the proof of Theorem 3.34 in this thesis.

❑

We have emphasized that the result holds for the *co-winner model* because our problem turns out to be hard when deciding whether or not c is the unique winner for at least one extension:

Theorem 3.12. *k-VETO-1TOS-POSSIBLE WINNER is NP-complete for every $k \geq 3$ under the unique-winner model.*

Proof. Our proof uses a reduction from 3DM. Let (X,Y,Z,E) be such an instance with $|X| = |Y| = |Z| = n$ and $|E| = m$. W.l.o.g., we assume that $m \geq n$ as otherwise a three-dimensional matching cannot exist. We show our result for 3-Veto and argue at the end of the proof how to adjust it for higher values k. Based on our 3DM instance (X,Y,Z,E), we construct the following instance of possible winner. Our candidate set is $C = \{c\} \dot\cup (X \dot\cup Y \dot\cup Z) \dot\cup F$, where $F := \{f_1, \ldots, f_{3m-3n}\}$ [7] and c is our distinguished candidate. We further have $\tilde{C} := C \setminus F$ as the totally ordered subset. Finally, there are m voters $v_1, \ldots, v_m$. Voter v_i vetoes the candidates according to triple $e_i = (x^i, y^i, z^i)$ such that $x^i \succ y^i \succ z^i$ holds, i.e., $x^i \in X$ is ranked third to last, $y^i \in Y$ second to last, and $z^i \in Z$ last among the candidates in $\tilde{C}$.

Note that $\mathit{vetoes}_{(\tilde{C},V)}(c) = 0$ and $\mathit{vetoes}_{(\tilde{C},V)}(a) = l(a)$ for each $a \in X \cup Y \cup Z$, where $l(a)$ denotes the number of triples in E containing a. Moreover, the candidates in F are not definitely vetoed by any voter. We show that there is a three-dimensional matching if and only if c is a possible unique winner.

($\Rightarrow$): Assume that there is a three-dimensional matching, i.e., there exists a subset of triples $E' \subseteq E$ such that each element in $X \cup Y \cup Z$ belongs to exactly one triple in E'. We construct the following completion of the partial profile:

- If $e_i \in E'$, voter v_i vetoes the candidates in e_i in our constructed extension. All candidates in F are ranked on better positions in v_i's complete ranking.

- For each $e_i \in E \setminus E'$, three vetoes go to candidates in F so that each of them is vetoed by exactly one voter. This is possible as there are $m-n$ voters not according to any triple in E' and these voters assign $3m-3n$ vetoes in total. Thus, the $3m-3n$ candidates $f_1, \ldots, f_{3m-3n}$ can be assigned one veto each.

[7] To avoid confusion, we point out that in many proofs, we follow the convention in the largest part of the COMSOC literature and assume that the symbols X, Y, and Z and the elements in these sets correspond to the original 3DM instance and at the same time stand for candidate sets and candidates in our possible winner instance. In some later proofs, e.g., both candidates in an election and vertices in a graph are denoted by the same symbol.

It follows that c is a unique winner with zero vetoes as each other candidate has one veto in our constructed profile completion.

($\Leftarrow$): Now presume that c is a possible unique winner. Since c has no vetoes for any extension, this requires that every non-distinguished candidate is vetoed at least once for our constructed extension. As m voters veto $3m$ candidates in total, and since there are $3n + (3m - 3n) = 3m$ candidates other than c, each of them must receive exactly one veto.

It remains to show that each voter either vetoes three or zero candidates in F in an extension c is the only winner with zero vetoes. To do so, we first regard candidates in Z. A necessary condition is that n vetoes (one for each candidate in Z) are left unchanged, i.e., are not replaced by some veto for any f_j. Thus, $m-n$ potential vetoes for candidates in Z must be replaced by vetoes for candidates in F. The same arguing can be applied—due to symmetry—for Y and X. However, as soon as one $f_j \in F$ is vetoed instead of some $z \in Z$ in a given vote v_i, this necessarily implies that v_i does not veto the other two candidates in the triple e_i either, that is, all three potential vetoes for candidates in $X \cup Y \cup Z$ go to candidates in F. This holds since voter v_i prefers x^i over y^i and y^i over z^i, and consequently z^i's veto can only fall away in case the other two vetoes fall away as well. As both for X, Y, and Z exactly $m-n$ vetoes are replaced and therefore n vetoes remain unchanged, there are n voters in total in our extension which assign $3n$ vetoes to candidates in $X \cup Y \cup Z$. Let $V = \{v_{i_1}, \ldots, v_{i_n}\}$ denote this set. Since each candidate in $X \cup Y \cup Z$ is vetoed by exactly one voter v_{i_h} ($1 \leq h \leq n$), the sets $e_{i_1}, \ldots, e_{i_n}$ form are three-dimensional matching.

The proof can be adjusted for $k \geq 4$ as follows. We introduce $(k-3) \cdot |V| = (k-3)m$ dummy candidates into the set $\tilde{C}$, each of them vetoed exactly once in subelection $(\tilde{C}, V)$, and each voter ranks exactly $k-3$ of them behind the candidates in $X \cup Y \cup Z$ in subelection $(\tilde{C}, V)$. The proof for 3-Veto can thus be directly applied as only the candidates in $X \cup Y \cup Z$ may lose some veto to some f_j, whereas all dummy candidates ranked behind must keep their vetoes in order to not tie with c. ❑

Theorem 3.11 and 3.12 present an instance of a problem where ties matter. Notice that in the co-winner model, c is trivially a possible winner whenever c is not vetoed in some extensions. Under the unique-winner model, zero vetoes for some extension suffice for c to be among the winners, but this does not mean that c is the only winner. Theorem 3.12 implicates that under the unique-winner model, possible winner for k-Veto, $k \geq 3$, is hard for the more general TOS model as well. Our next result shows that the TOS model does not make a difference between the unique- and co-winner model for the 3-Veto rule. Actually, both models lead to a hard possible winner problem.

Theorem 3.13. 3-VETO-TOS-POSSIBLE WINNER *is* NP*-complete under the co-winner model.*

Proof. In order to prove our result in the co-winner case, we reduce from 3DM. Based on a 3DM instance (X, Y, Z, E) with $|X| = |Y| = |Z| = n$, $|E| = m$, and $m \geq n$ (otherwise, a matching does not exist), our 3-VETO-TOS-POSSIBLE WINNER instance is defined as follows. Our candidate set is defined by $C = \{c, g_1, g_2\} \dot{\cup} X \dot{\cup} Y \dot{\cup} Z \dot{\cup} F$, where $F := \{f_1, \ldots, f_{3m-3n}\}$. Moreover, c is our designated candidate. The voter set contains $|E| + 1$ voters defined as follows.

1. Voter v_i votes $(C^{v_i} \setminus \{x^i, y^i, z^i\}) \succ x^i \succ y^i \succ z^i$, where $C^{v_i} = C \setminus F$ and $e_i = (x^i, y^i, z^i)$ (with $x^i \in X$, $y^i \in Y$, $z^i \in Z$, and $1 \leq i \leq m$). All candidates in C^{v_i} not vetoed are ranked in arbitrary, but fixed order.

2. Voter v_{m+1} definitely vetoes c, g_1, and g_2, in arbitrary order. We have $C^{v_{m+1}} = C$. All candidates not vetoed are ranked arbitrarily.

We show that there is a three-dimensional matching if and only if c is a possible winner.

($\Rightarrow$): Let $E' \subseteq E$ be a three-dimensional matching. We construct the following profile completion:

- If $e_i \in E'$, voter v_i in the first group is meant to veto the candidates in e_i, i.e., x^i, y^i, and z^i.
- If $e_i \in E \setminus E'$, voter v_i vetoes three candidates in F such that each f_j ($1 \leq j \leq 3m-3n$) is vetoed exactly once.

It follows that all candidates in C, including c, are winners with one veto each.

($\Leftarrow$): Assume that c is a winner for at least one extension. Since c has one definite veto, each other candidate must have at least one veto. For g_1 and g_2, this holds as both are definitely vetoed by voter v_{m+1}. The voters $v_1, \ldots, v_m$ assign precisely $3m$ vetoes to all candidates in $F \cup X \cup Y \cup Z$. Since the cardinality of this set is exactly $3m$, in our extension of the partial profile all candidates in $F \cup X \cup Y \cup Z$ have exactly one veto each.

It remains to show that the only way to ensure this is the existence of a three-dimensional matching. Since the candidates in X, Y, and Z receive the same numbers of potential vetoes, and since exactly n vetoes must be considered for each of the three candidate groups, we find that exactly $m-|X| = m-|Y| = m-|Z| = m-n$ vetoes for each of the three groups X, Y, and Z must be replaced with vetoes for candidates in F. Analogously to the arguing in the proof of Theorem 3.12, it follows that in an extension with co-winner c, the candidates vetoed by the first m voters must be either three candidates in F or three candidates in $X \cup Y \cup Z$ for our constructed completion. Let $v_{i_1}, \ldots, v_{i_n}$ ($1 \leq i_1 < \ldots < i_n \leq m$) be the n voters vetoing three candidates in $X \cup Y \cup Z$ in our extension. Similarly to the arguing in the proof of Theorem 3.12, it follows that $e_{i_1}, \ldots, e_{i_n}$ form a three-dimensional matching of $X \cup Y \cup Z$. ❑

So far we have seen that 1TOS yields three easiness results for k-Veto ($k \leq 3$) in the co-winner model. Next we prove that our problem is hard for $k \geq 4$ in the co-winner case.

Theorem 3.14. *k-VETO-1TOS-POSSIBLE WINNER is NP-complete for $k \geq 4$ under the co-winner model.*

Proof. First we prove hardness for 4-Veto. At the end of the proof, we point out how to adjust the construction for higher values of k. We show hardness by reducing from R3DM. Remember that in a R3DM instance (X,Y,Z,E) with $|X| = |Y| = |Z| = n$ and $E = \{e_1, \ldots, e_m\}$, each element in $X \cup Y \cup Z$ occurs in exactly two or three triples in E. In particular, it holds $2n \leq m \leq 3n$. The two extremes correspond to the cases where every a belongs to exactly two and three triples, respectively. Additionally, we assume that $n > 1$ which is no actual restriction to R3DM as otherwise the problem is trivial. In summary, we have $3n \geq m \geq 2n > n > 1$. We create the following election (C,V). The candidate set is $C = \{c, f\} \dot\cup X \dot\cup Y \dot\cup Z \dot\cup D \dot\cup B \dot\cup G$, where $D := \{d_1, \ldots, d_5\}, B := \{b_1, b_2, b_3\}$, $G := \{g_1, \ldots, g_{\lfloor \frac{3n(m-n-1)}{m-n} \rfloor}\}$, and c is the designated candidate. Each voter's totally ordered subset $\tilde{C}$ is defined as $\tilde{C} := C \setminus B$. There are the following three voter groups in V:

1. There are $m-n$ voters u_i ($i=1,\ldots,m-n$) voting $\tilde{C}\setminus\{d_1,d_2,d_3,c\} \succ d_1 \succ d_2 \succ d_3 \succ c$.

2. There are m voters v_i ($1 \leq i \leq m$), where voter v_i votes $\tilde{C}\setminus\{x^i,y^i,z^i,d_4\} \succ x^i \succ y^i \succ z^i \succ d_4$ (we have $x^i \in X$, $y^i \in Y$, $z^i \in Z$, and $e_i = (x^i,y^i,z^i)$ denotes the ith triple in E).

3. The third group contains exactly $3n(m-n-1)$ voters with the following properties:

   - For each $j \in \{1,\ldots,\lfloor\frac{3n(m-n-1)}{m-n}\rfloor\}$, there are $m-n$ voters w_i in this group who rank g_j fourth-to-last among the candidates in $\tilde{C}$.
   - The remaining $3n(m-n-1)-(m-n)\cdot\lfloor\frac{3n(m-n-1)}{m-n}\rfloor = 3n(m-n-1) \mod (m-n) (\in \{0,1,\ldots,m-n-1\})$ voters rank f fourth-to-last. (This number is equal to zero in case $m-n$ divides $3n(m-n-1)$.) W.l.o.g., these are the voters w_i with largest index.
   - For each $a \in X \cup Y \cup Z$ there are exactly $m-n-1$ voters w_i ranking a third-to-last in $(\tilde{C},V)$.
   - Exactly $(m-n)-((3n(m-n-1)) \mod (m-n)) (\leq m-n)$ voters rank f on second-to-last position in subelection $(\tilde{C},V)$. W.l.o.g., these are the first voters ranking g_1 on the fourth-to-last position restricted to the candidates in $\tilde{C}$. [8]
   - All remaining voters rank d_4 second to last in subelection $(\tilde{C},V)$.
   - Each voter w_i ranks d_5 last in subelection $(\tilde{C},V)$.

The construction is built in a way that b_1, b_2, and b_3 may only be vetoed in votes from the second voter group, i.e., the only unsure vetoes that may fall away in favor of these three candidates are vetoes for candidates in $X \cup Y \cup Z$ in the second voter group. The first voter block fixes the number of definite vetoes for c. There are three other candidates in $\tilde{C}$ possibly but not definitely vetoed by these voters, but these vetoes must remain unchanged for extensions c is supposed to win. The third voter group assigns the required vetoes to all candidates in $X \cup Y \cup Z$. Note that f and candidates in G all have the same number of potential vetoes as c. Hence, none of their vetoes may fall away. Although candidates in $X \cup Y \cup Z$ get further vetoes from voters in the third group, these vetoes must not be replaced as otherwise f or candidates in G lose their vetoes as well and thus one of them beats c. One might regard the vetoes for the candidates in G and f (i.e., the fourth to last positions in subelection $(\tilde{C},V)$ in the third voter group) as a wall of vetoes that must not be substituted, and candidates in $X \cup Y \cup Z$ therefore must not lose their vetoes from this group either. Tables 3.4 and 3.3 visualize two concrete instances with $n=2$, $m=5$ and $n=2$, $m=6$, respectively.

In subelection $(\tilde{C},V)$, the veto numbers are as follows:

$vetoes_{(\tilde{C},V)}(c) = vetoes_{(\tilde{C},V)}(c') = m-n$ for each $c' \in \{d_1,d_2,d_3,g_1,\ldots,g_{\lfloor\frac{3n(m-n-1)}{m-n}\rfloor},f\}$, $vetoes_{(\tilde{C},V)}(a) = m-n-1+l(a)$ for $a \in X \cup Y \cup Z$ (where $l(a)$ denotes the number of triples in E containing a), $vetoes_{(\tilde{C},V)}(d_4) \geq m > m-n$ (this holds due to vetoes from the second voter group) and $vetoes_{(\tilde{C},V)}(d_5) = 3n(m-n-1) \geq m-n$. [9] Thus, c is a winner in subelection $(\tilde{C},V)$.

[8] Hence, there are no overlap in a sense that the same voter ranks f fourth-to-last and second-to-last in subelection $(\tilde{C},V)$.

[9] We can transform this inequality into the equivalent inequality $m \geq n(2+3n)/(3n-1)$. As $m \geq 2n$ holds (recall from above that there are at least $2n$ triples in E for a given instance of R3DM), it follows that $n(2+3n)/(3n-1) \leq 2n (\leq m)$ which holds for $n \geq 4/3$ or, as $n \in \mathbb{N}$, for $n > 1$.

We claim that a three-dimensional matching exists if and only if c is a possible winner.

($\Rightarrow$): Suppose that a matching exists. We construct an extension as follows. b_1, b_2, and b_3 are ranked last, second-to last and third-to-last in each vote v_i of the second group for which the corresponding triple (x^i, y^i, z^i) is not in the matching. By this, we obtain the final veto numbers $vetoes(c') = m - n$ for each $c' \in B \cup G \cup X \cup Y \cup Z \cup \{c, d_1, d_2, d_3, f\}$. The reason is that $l(a) - 1$ vetoes for $a \in X \cup Y \cup Z$ in the second voter group are replaced by vetoes for candidates in B and one veto in the second group is left unchanged for each a). Moreover, we have $vetoes(d_4) > m > m - n$ (because of m definite vetoes from the second voter group and some more vetoes from the third voter group) and $vetoes(d_5) = 3n(m - n - 1) \geq m - n$. Since no candidate has fewer vetoes than c in our constructed extension, c is a winner for this extension and thus possible winner.

($\Leftarrow$): Presume that c is a possible winner. As mentioned above, c is a winner in the partial election $(\tilde{C}, V)$ with $m - n$ definite vetoes from the first voter group. Unfortunately, the candidates in B require $m - n$ vetoes each in the overall election for some extension c is supposed to win. Thus, for this extension, they must be enqueued into the partial election $(\tilde{C}, V)$ in a way that some other candidate being vetoed in the subelection is not vetoed in the entire election (C, V). Our proof exploits the fact that in case a candidate on fourth-to-last position in a vote (among the candidates in $\tilde{C}$) must not be replaced by a veto for some candidate in B, no other veto can be replaced in the same vote since replacing the candidate on the third- or second-to-last position (restricted to the candidates in $\tilde{C}$) requires that the veto on the fourth-to-last position is replaced as well. Moreover, last positions in subelection $(\tilde{C}, V)$ never fall away.

Observe that for an extension with co-winner c, all vetoes in the first group must remain unchanged: d_1, d_2, and d_3 have the same veto number as c and according to this must not lose any veto. The same holds for group three: Neither any g_j nor f may lose any veto since they all are tied with c when we account for all their potential vetoes. Unfortunately, in the third voter group each voter must veto his fourth-to-least preferred candidate in $\tilde{C}$ as these vetoes either go to f or some g_j. Hence, in an extension with winner c, voters in the third group veto four candidates in $\tilde{C}$ each.

This implies that each candidate in B receives $n - m$ vetoes from voters in the second group. Thus, b_1, b_2, and b_3 must replace $3m - 3n$ vetoes for candidates in $X \cup Y \cup Z$ in total.

Voters in the second group assign $3m$ unsure vetoes to non-distinguished candidates. As each candidate in $X \cup Y \cup Z$ is potentially vetoed by $m - n - 1$ voters in the third group, each of them needs at least one additional veto from voters in the second group. Otherwise, at least one of them beats c. Moreover, the three candidates in B need $3m - 3n$ vetoes in total from voters of group two. Hence, the candidates in $B \cup X \cup Y \cup Z$ require at least $3m$ additional vetoes in total. As a combined result, voters from the second group assign one veto to each candidate in $X \cup Y \cup Z$ provided that c is a winner for our constructed extension.

Arguing similarly to the proof of Theorem 3.12, it follows that in our constructed completion with winner c each voter in the second group either vetoes three candidates in $X \cup Y \cup Z$ or three candidates in B. According to this, $m - n$ of these voters assign $3m - 3n$ vetoes to the candidates in B and n of these voters assign $3n$ vetoes to candidates in $X \cup Y \cup Z$. Assuming that each of the n voters $v_{i_1}, \ldots, v_{i_n}$ vetoes three candidates in $X \cup Y \cup Z$, and since each $a \in X \cup Y \cup Z$ is vetoed by exactly one such voter (otherwise, some candidate in $X \cup Y \cup Z$ beats c), the sets $e_{i_1}, \ldots, e_{i_n}$ necessarily form a three-dimensional matching.

For higher values k, we simply add $k - 4$ dummy candidates ranked on the last $k - 4$ positions in subelection $(\tilde{C}, V)$ by every voter. The vetoes of these padding candidates are fixed. ❑

First voter group				
Voter	4	3	2	1
u_1	d_1	d_2	d_3	c
u_2	d_1	d_2	d_3	c
u_3	d_1	d_2	d_3	c
u_4	d_1	d_2	d_3	c

Second voter group				
Voter	4	3	2	1
v_1	x_1	y_1	z_1	d_4
v_2	x_1	y_2	z_2	d_4
v_3	x_1	y_1	z_2	d_4
v_4	x_2	y_2	z_1	d_4
v_5	x_2	y_1	z_1	d_4
v_6	x_2	y_2	z_2	d_4

Third voter group				
Voter	4	3	2	1
w_1	g_1	x_1	f	d_5
w_2	g_1	x_1	f	d_5
w_3	g_1	x_1	d_4	d_5
w_4	g_1	x_2	d_4	d_5
w_5	g_2	x_2	d_4	d_5
w_6	g_2	x_2	d_4	d_5
w_7	g_2	y_1	d_4	d_5
w_8	g_2	y_1	d_4	d_5
w_9	g_3	y_1	d_4	d_5
w_{10}	g_3	y_2	d_4	d_5
w_{11}	g_3	y_2	d_4	d_5
w_{12}	g_3	y_2	d_4	d_5
w_{13}	g_4	z_1	d_4	d_5
w_{14}	g_4	z_1	d_4	d_5
w_{15}	g_4	z_1	d_4	d_5
w_{16}	g_4	z_2	d_4	d_5
w_{17}	f	z_2	d_4	d_5
w_{18}	f	z_2	d_4	d_5

Table 3.3: Example of the instance in the proof of Theorem 3.14 for $n = 2$, $m = 6$. The column labeled with "h" presents each voter's candidate ranked on position $|\tilde{C}| + 1 - h$ in subelection $(\tilde{C}, V)$ ($h = 1, \ldots, 4$). The R3DM instance (X, Y, Z, E) we reduce from is defined by $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$, $Z = \{z_1, z_2\}$, and $E = \{e_1, \ldots, e_6\}$ with $e_1 = (x_1, y_1, z_1)$, $e_2 = (x_1, y_2, z_2)$, $e_3 = (x_1, y_1, z_2)$, $e_4 = (x_2, y_2, z_1)$, $e_5 = (x_2, y_1, z_1)$, and $e_6 = (x_2, y_2, z_2)$. Notice that, for instance, e_1 and e_6 form a three-dimensional matching.

First voter group				
Voter	4	3	2	1
u_1	d_1	d_2	d_3	c
u_2	d_1	d_2	d_3	c
u_3	d_1	d_2	d_3	c

Second voter group				
Voter	4	3	2	1
v_1	x_1	y_2	z_1	d_4
v_2	x_1	y_1	z_1	d_4
v_3	x_2	y_2	z_2	d_4
v_4	x_2	y_1	z_2	d_4
v_5	x_2	y_1	z_1	d_4

Third voter group				
Voter	4	3	2	1
w_1	g_1	x_1	f	d_5
w_2	g_1	x_1	f	d_5
w_3	g_1	x_2	f	d_5
w_4	g_2	x_2	d_4	d_5
w_5	g_2	y_1	d_4	d_5
w_6	g_2	y_1	d_4	d_5
w_7	g_3	y_2	d_4	d_5
w_8	g_3	y_2	d_4	d_5
w_9	g_3	z_1	d_4	d_5
w_{10}	g_4	z_1	d_4	d_5
w_{11}	g_4	z_2	d_4	d_5
w_{12}	g_4	z_2	d_4	d_5

Table 3.4: Example of the instance in the proof of Theorem 3.14 for $n = 2$, $m = 5$. The column labeled with h presents each voter's candidate ranked on position $|\tilde{C}| + 1 - h$ in subelection $(\tilde{C}, V)$ ($h = 1, \ldots, 4$). By way of example, the column labeled with "1" contains all voters' least preferred candidates in $\tilde{C}$. The R3DM instance (X, Y, Z, E) we reduce from is defined by $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$, $Z = \{z_1, z_2\}$, and $E = \{e_1, \ldots, e_5\}$ with $e_1 = (x_1, y_2, z_1)$, $e_2 = (x_1, y_1, z_1)$, $e_3 = (x_2, y_2, z_2)$, $e_4 = (x_2, y_1, z_2)$, and $e_5 = (x_2, y_1, z_1)$. Observe that, e.g., e_1 and e_4 form a three-dimensional matching.

Finally, we mention that the complexity of 2-APPROVAL-TOS-POSSIBLE WINNER and 2-VETO-TOS-POSSIBLE WINNER is still open.

3.5 Necessary Bribery

In this section, we investigate the impact of partial information on the complexity of bribery. More precisely, we consider a hybridization of standard bribery and necessary winner. Unless stated otherwise, the results in this section can be found in [30]. The necessary bribery problem is defined as follows.

$\mathscr{F}$-X-NECESSARY BRIBERY	
Given:	An election (C,V) with a set C of m candidates, a set V of n voters according to $X \in \overline{\text{PIM}}$, a designated candidate $c \in C$, and a nonnegative integer ℓ.
Question:	Is it possible to change up to ℓ votes such that c is a winner of the election under $\mathscr{F}$ no matter how we complete the votes in V?

In other words, we ask whether the briber can make c a necessary winner by changing at most ℓ votes in V. Note that each bribed vote is complete after the bribery, i.e., the briber can determine the complete ranking of every bribed voter according to his wishes. Nevertheless it would be an interesting task for future research to investigate what happens when bribed voters are irresolute in a way that their rankings after the bribery are—to a certain extent—incomplete. This applies to real-word applications where the briber's effort to convince some of the voters does not necessarily result in a complete change of the voters' opinions (rankings in this context), and the preferences of the bribed voters are still undetermined in a sense (cf. [54, 150] for some restricted bribery variants where, loosely speaking, the briber can only alter parts of a voter's ranking in general) In other words, the briber could achieve a certain change in a voter's opinion, but he cannot be secure that the voter will actually vote in accordance with the briber's preferences. In our setting, however, we premise that the briber can entirely alter a voter's ranking.

To avoid confusion, we point out that we dropped the word "necessary" in the original problem definition in [30] as the necessary winner variant appeared to us as the standard way to define bribery under partial information the more so as the briber wants to be sure to reach his goal and make c a winner for every completion of the unbribed voters' votes. In this thesis, we make use of the term "necessary bribery" in order to emphasize that necessary bribery is a combination of necessary winner and standard bribery. Likewise, possible bribery, studied in Section 3.6, is defined as a hybridization of possible winner and standard bribery. By saying "necessary bribery" instead of "bribery", we keep our notation consistent. Table 3.5 indicates the results on the complexity of necessary bribery under partial information in k-Approval and k-Veto.

In this and the following section (about possible bribery), if not mentioned other, our input is an election (C,V) with m candidates in candidate set $C = \{c_1,\dots,c_{m-1},c\}$, n voters in voter set $V = \{v_1,\dots,v_n\}$ (with $m \in \mathbb{N}$, $n \in \mathbb{N}_0$), a designated candidate $c \in C$, and a nonnegative integer ℓ, the bribery limit. We assume that all votes in V are partial according to a given model X in PIM. Besides, $(C,\bar{V})$ denotes the final election after the bribery, that is, $\bar{V}$ is identical to V in the votes not bribed, but instead contains the bribed voters' new complete rankings. Again, we sometimes make use of the symbol d to denote an arbitrary (non-distinguished) candidate. When studying the 1TOS

Voting rule	FI	Gaps	FP	TOS	PC	CEV	1TOS	1Gap	TTO	BTO
Plurality	P	**NPC**	**NPC**	**NPC**	**NPC**	**P**	**P**	**NPC**	**P**	**NPC**
2-Approval	P	**NPC**	**NPC**	**NPC**	**NPC**	**P**	**P**	**NPC**	**P**	**NPC**
k-Approval ($k \geq 3$)	NPC	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*
Veto	P	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
2-Veto	P	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
3-Veto	P	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
k-Veto ($k \geq 4$)	NPC	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*

Table 3.5: Summary of the complexity results for necessary bribery under the co-winner model. Key: NPC stands for "NP-complete", P stands for "membership in P". Column FI displays the results for the case with full information due to Faliszewski et al. [79] and Lin [119]. Results in italic are hardness results that follow from already existing hardness results for full information. Results in boldface are new.

model, we denote candidates in $\tilde{C}$ by c' and candidates in $C \setminus \tilde{C}$ by $p_1, p_2, \ldots$, by p (if $C \setminus \tilde{C} = \{p\}$), or by p and q (if $|C \setminus \tilde{C}| = 2$). Analogously to the previous section, we assume that $k < m$ when studying k-Approval or k-Veto. Last but not least, unless stated otherwise, we show our results under the co-winner model.

Our first theorem states that necessary bribery for a model X is at least as hard as bribery under full information and necessary winner for model X.

Theorem 3.15. *$\mathscr{F}$-FI-BRIBERY and $\mathscr{F}$-X-NECESSARY WINNER many-one reduce to $\mathscr{F}$-X-NECESSARY BRIBERY for any model $X \in$ PIM and voting rule $\mathscr{F}$.*

Proof. The first reduction holds since every model in PIM is a generalization of FI, i.e., there are instances where each vote is complete. The other reduction follows because necessary bribery with bribery limit $\ell = 0$ coincides with necessary winner given partial model X. ❑

Theorem 3.15 can be applied to derive some hardness results either from necessary winner or from bribery under full information. According to Section 3.4, necessary winner is easy even for every scoring rule.

Our first complexity result says that, although bribery in Plurality and 2-Approval is easy under full information, there are some partial information models for which the complexity of bribery increases.

Theorem 3.16. *k-APPROVAL-X-NECESSARY BRIBERY is NP-complete for $k \leq 2$ and $X \in \{Gaps, 1Gap, FP, TOS, PC, BTO\}$.*

Proof. It suffices to show our result for TOS and BTO since all the remaining structures in X inherit the NP-hardness lower bound from TOS or BTO. We first show our result for Plurality and argue at the end of the proof how to adjust the proof for 2-Approval. In order to show hardness, we reduce from X3C. Based on an X3C instance $(B, \mathscr{S})$, with $B = \{b_1, \ldots, b_{3m}\}$ and $\mathscr{S} = \{S_1, \ldots, S_n\}$ (with $S_i \subset B$ and $|S_i| = 3$ for each $i \in [n]$, and w.l.o.g. $n \geq m$ as otherwise an exact cover is impossible), we construct our necessary bribery instance as follows. There are $3m+1$ candidates in candidate set $C = B \dot{\cup} \{c\}$ where c is our designated candidate. Moreover, the briber may bribe up

to $\ell = m$ voters. The set of voters V consists of $3mn - n + 2m$ voters. [10] The partial profile P is defined as follows.

1. For each i, $1 \leq i \leq n$, there is one voter v_i possibly but not definitely approving of the candidates in S_i. Given the BTO model, we have $C_T^{v_i} = S_i$, $C_B^{v_i} = C \setminus S_i$, and v_i arbitrarily ranks the candidates in the bottom set. For the TOS model, we construct the votes v_i as follows. Suppose that $S_i = \{b_1^i, b_2^i, b_3^i\}$ contains the candidates potentially ranked first by voter v_i, and let the remaining candidates be certainly disapproved by this voter. We set $C^{v_i} = C \setminus \{b_2^i, b_3^i\}$ as the totally ordered subset of candidates ranked by voter v_i and assume that $b_1^i \succ (C \setminus S_i)$ holds in voter v_i's ranking of the candidates in C^{v_i}. This makes sure that exactly the candidates in S_i are the potential scorers of the vote v_i.

2. For each j, $1 \leq j \leq 3m$, there are $n + 1 - l_j$ complete votes ranking b_j first, where $l_j = |\{S_i \in \mathscr{S} : b_j \in S_i\}|$.

3. There are $n - m$ complete votes favoring c.

Recall that complete votes can be displayed by every model in PIM, in particular by TOS and BTO.

Note that $score(c) = n - m$ and $pscore(b_j) = n + 1$, for all $j \in [3m]$. We claim that there exists an exact cover of three-sets if and only if c can be made a winner of the election in all completions.

($\Rightarrow$): Assume there is an exact cover $\mathscr{S}' \subseteq \mathscr{S}$, i.e., for every $b_j \in B$ exists exactly one $S_i \in \mathscr{S}'$ such that $b_j \in S_i$. Suppose that $\mathscr{S}' := \{S_{i_1}, \ldots, S_{i_m}\}$, where $S_{i_h} \in \mathscr{S}$ for every $h \in [m]$ and the S_{i_h} are pairwise different. By changing the votes v_{i_h} in the first voter group (with the candidates in S_{i_h} as possible scorers) and by setting c on the first place in each of these votes, the briber ensures that $score_{(C,\bar{V})}(c) = n \geq pscore_{(C,\bar{V})}(b_j) = n$ for each $b_j \in B$ which implies that c is a winner of the election under all possible complete profiles after the bribery. Thus, c is a necessary winner.

($\Leftarrow$): Suppose that the briber can make c a winner of the election in all possible completions by bribing at most m voters. Since c cannot reach a score of $n + 1$, each b_j has to lose at least one potential point. This is only possible if these m votes cover $3m$ candidates, and thus possible points. This in turn requires that the briber bribes m voters $v_{i_1}, \ldots, v_{i_m}$ of the first voter group and the sets $S_{i_1}, \ldots, S_{i_m}$ exactly cover B. Otherwise, there is some $b_j \in B$ not belonging to any S_{i_h} ($h \in [m]$) and thus we obtain $pscore_{(C,\bar{V})}(b_j) = n + 1 > score_{(C,\bar{V})}(c) = n$. As there are extensions where b_j receives his maximum possible score, c does not win in these extensions and is hence not a necessary winner after the bribery.

As pointed out above, the proof for 2-Approval makes use of a similar construction. First of all, there are $3mn + 3m - 2n$ additional dummy candidates $d_1, \ldots, d_{3mn+3m-2n}$ such that the first $3mn + 2m - 2n$ dummy candidates (i.e., the ones with the smallest index) are potentially approved of by exactly one voter each and the remaining dummy candidates do not score at all in the initial election. Again, there are $3mn - n + 2m$ voters, divided into three groups. The votes in the first voter group are defined as follows:

[10] This number is always nonnegative: $3mn - n + 2m \geq 0$ if and only if $m(3n+2) \geq n$ if and only if $m \geq \frac{n}{3n+2}$. Since $\frac{n}{3n+2} < 1$ is equivalent to $n > -1$, we can deduce $m \geq 1 > \frac{n}{3n+2}$. Thus, the inequality holds for all $m, n \in \mathbb{N}$ and in particular for $n \geq m$.

- Given the BTO model, we again have $C_T^{v_i} = S_i$ and $C_B^{v_i} = C \setminus S_i$. The candidates in the bottom set are arbitrarily ranked.

- In the TOS model, the voters v_i, $1 \leq i \leq n$ are defined as follows. We set $S_i = \{b_1^i, b_2^i, b_3^i\}$ and the vote v_i is represented by the totally ordered subset $C^{v_i} = C \setminus \{b_3^i\}$. We set $pos_{C^{v_i}}(b_j^i, v_i) = j$ for $j = 1, 2$. The other candidates in C^{v_i} are arbitrarily ranked behind b_1^i and b_2^i. This way, exactly the candidates from S_i are the potentially but not necessarily definitely approved candidates in this vote. [11] The remaining candidates are certainly disapproved by v_i.

Once again, all voters from the second and third group are complete and approve of the same candidates in $B \cup \{c\}$ as before plus one of the $3mn + 2m - 2n$ first dummy candidates such that each of them is potentially approved exactly once. The remainder of the proof works analogously to the proof for the Plurality rule: the briber bribes the voters according to the exact cover and makes each voter approve of c and the m dummy candidates not scoring in the original election, that is, the ones with the largest index. ❑

Observe that the proof uses a somewhat similar construction to the proof in [79] showing that APPROVAL-BRIBERY (that is, under complete information) is NP-complete.

The subsequent results reveal that necessary bribery under the remaining structures is easy in Plurality.

Theorem 3.17. PLURALITY-TTO-NECESSARY BRIBERY *and* PLURALITY-CEV-NECESSARY BRIBERY *are in* P.

Proof. We only have to show our result for TTO as CEV is a special case of TTO. Roughly, we can divide the votes V into the following subsets:

1. $V_0 := \{v \in V : C_T^v = \emptyset\}$ as the votes with empty top set and
2. $V_1 := V \setminus V_0 = \{v \in V : C_T^v \neq \emptyset\}$.

Voters in V_1 have a unique top candidate and thus—albeit possibly incomplete—assign a unique score to each candidate in C. We may therefore w.l.o.g. assume that these voters completely rank the candidates in C, by arbitrarily ranking the bottom set candidates. In contrast, votes in V_0 may have each candidate in C as their top choice. Since the briber wants to make c a winner for each extension of these votes, we presume that none of them likes c most (which is possible for some completions of V_0). From this arguing, we obtain the scores $score(c) = score_{(C,V_1)}(c)$ and $pscore(c_j) = score_{(C,V_1)}(c_j) + |V_0|$ ($j \in [m-1]$).

A rational briber bribes only voters not definitely voting for c and each bribed voter favors c after the bribery. Observe also that the briber should bribe as many V_0 voters as possible. Bribing a voter in V_0, the briber decreases the potential score of each non-distinguished candidate by one, whereas bribing a voter definitely favoring some c_j only decreases this value for c_j, but not for the other candidates different from c. (Note that for $|C| = 2$, there is no need to distinguish between V_1

[11] We point out that voter v_i certainly approves of b_1^i and possibly but not definitely approves of one of the candidates b_2^i and b_3^i. Since each non-distinguished candidate may obtain all potential approvals for some completions, it is immaterial to our analysis whether or not a potential approval for a candidate in $C \setminus \{c\}$ is certain for a given vote.

and V_0. Every voter in V_1 either votes $c \succ c_1$ or $c_1 \succ c$ (where $C = \{c, c_1\}$), while every voter in V_0 votes $c_1 \succ c$, at least in c's worst-case. According to this, for two candidates the votes in V_0 are complete in c's worst-case in a sense).

Our algorithm checks the following three cases:

- $score(c) + \ell \geq \frac{n}{2}$. Our instance is a YES instance as for every extension the briber can ensure that at least half of all voters in V definitely prefer c to all other candidates. Since all other candidates can reach at most half of all points, c is surely a necessary winner in the final election, no matter how we complete the partial, unchanged votes.

- $score(c) + \ell < \frac{n}{2}$ and $|V_0| \geq \ell$. Then the briber bribes ℓ voters in V_0 and makes every bribed voter rank c first. We accept if and only if the following condition holds:

$$score(c) + \ell \geq pscore(c_j) - \ell \quad \forall j \in [m-1].$$

  According to this inequality, c has at least $score(c) + \ell$ (definite) points in the final election $(C, \bar{V})$ and there is not any c_j whose potential score (including all definite and uncertain points that c_j can reach in the final election for any extension) exceeds the final score of c.

- $score(c) + \ell < \frac{n}{2}$ and $|V_0| < \ell$. Then the briber bribes all voters in V_0 and $\ell - |V_0|$ voters in V_1. Note that the changed votes in V_0 are complete after the bribery (with top candidate c) and—by completing arbitrarily all votes in V_1 (which does not change the score of any candidate)—we obtain an equivalent bribery problem under full information with bribery limit $\tilde{\ell} := \ell - |V_0|$ and new voter set $\tilde{V} := V_1 \cup \tilde{V}_0$ (where $\tilde{V}_0$ contains $|V_0|$ voters ranking c first and V_1 contains all voters with non-empty top set from the original election (C, V)). From [79], we know that bribery under full information is easy. Thus, this subcase is in P as well.

Observe that determining the scores and the first two cases can be done in $O(n \cdot m)$ time. In the third subcase, we can regard an equivalent bribery instance under full information with candidate set C, voter set $\tilde{V}$, distinguished candidate c, and bribery limit $\tilde{\ell}$. As already mentioned, bribery under complete information is easy. ❑

We now turn to the 1TOS model.

Theorem 3.18. PLURALITY-1TOS-NECESSARY BRIBERY *is in* P.

Proof. We let $\tilde{C} \subseteq C$ denote the candidates who are completely ranked from first to last by all voters. Our algorithm distinguishes the following cases:

- $\tilde{C} = C$. This case corresponds to bribery under full information which is known to be in P [79].

- $(C \setminus \tilde{C}) \setminus \{c\} \neq \emptyset$. In other words, there is at least one non-distinguished candidate p in $C \setminus \tilde{C}$. Since there are extensions where p is ranked first by every voter, we have $score_{(C,V)}(c) = 0$ (the definite score of c before the bribery) and $score_{(C,\bar{V})}(c) = \ell$ (the number of definite points for c after the bribery). As $pscore_{(C,V)}(p) = n$ and $pscore_{(C,\bar{V})}(p) = n - \ell$ holds for each $p \in C \setminus \tilde{C}$, a necessary condition for c to catch all other candidates $p \notin \tilde{C}$ is $\ell \geq \frac{n}{2}$. Otherwise,

there are extensions of the final election where p has more points than c. Observe that $\ell \geq \frac{n}{2}$ is also sufficient since no other candidate $d \neq c$ can have $pscore_{(C,\bar{V})}(d) > \ell \geq \frac{n}{2}$ potential points after the bribery.

- $C \setminus \tilde{C} = \{c\}$. In this subcase, there are extensions where c is ranked last by every voter, that is, we have $score_{(C,V)}(c) = 0$ and $pscore_{(C,V)}(d) = score_{(\tilde{C},V)}(d)$ $(d \in C \setminus \{c\})$ in the original election. Loosely speaking, we regard an election under full information where each voter ranks c last (or at least not on first place). We again refer to the polynomial-time algorithm for bribery in Plurality in [79].

It follows that our problem is in P as the scores can be computed in $O(mn)$ time, the first and third subcase can be easily reduced to bribery under full information (which is known to be easy), and the second subcase requires the validation of one polynomially bounded inequality. ❑

In a nutshell, although bribery in Plurality is easy under complete information, six out of nine models of partial information raise the complexity. Only under the models CEV, TTO, and 1TOS we can decide our problem in polynomial time. This is somewhat astonishing since Plurality has a very simple inherent structure. Notice that BTO and TTO yield—although very similar structures—different complexities. Also note that our problem is easy for 1TOS, but hard for BTO, Gaps, 1Gap, and FP—four models for which possible winner turned out to be much easier than for the 1TOS structure, applied to the k-Approval/-Veto families.

Next we show that necessary bribery in 2-Approval is easy for the same three models as Plurality. Let us first regard the TTO and CEV structures.

Theorem 3.19. *Both* 2-APPROVAL-TTO-NECESSARY BRIBERY *and* 2-APPROVAL-CEV-NECESSARY BRIBERY *are in* P.

Proof. We only give the proof for the TTO structure, this immediately implies that 2-APPROVAL-CEV-NECESSARY BRIBERY is in P too.

Let V_0 denote the set of voters with an empty top set, that is $V_0 := \{v \in V : C_T^v = \emptyset\}$. Furthermore, let $V_1^c := \{v \in V : C_T^v = \{c\}\}$, $V_1^{\neg c} := \{v \in V : |C_T^v| = 1, c \notin C_T^v\}$, $V_2^c := \{v \in V : |C_T^v| \geq 2, c \in C_T^v, pos(c,v) \leq 2\}$, and $V_2^{\neg c} := \{v \in V : |C_T^v| \geq 2, pos(c,v) > 2\}$. In a similar manner, we define $V_i := V_i^c \cup V_i^{\neg c}$ $(i = 1,2)$. In other words, V_1 embraces all voters with singleton top set, whereas V_2 includes all voters with two or more candidates in their top set. Moreover, V_i^c $(V_i^{\neg c})$ includes all voters in V_i (not) definitely approving of c $(i = 1,2)$.

By means of these sets, we are equipped to determine the score of c and the potential scores of each c_j $(j \in [m-1])$:

$$score_{(C,V)}(c) = |V_1^c \cup V_2^c|, \quad pscore_{(C,V)}(c_j) = |V_0 \cup V_1| + score_{(C,V_2)}(c_j).$$

Note that all votes with at least two candidates in their top set give us all the information we need, i.e., we can uniquely determine the scores these voters assign to each candidate—no matter how we actually complete these votes. Votes with only c in the top set give a definite approval to c, while each c_j is potentially approved by these voters. The votes in $V_0 \cup V_1^{\neg c}$ give a potential point to each c_j, but surely no point to c (in the worst-case of c).

Observe that a reasonable briber makes each bribed voter approve of c after the bribery. The question arises which voters should be bribed, which ones not, and which ones with highest priority.

To find an answer to this question, we next check what c gains compared to the other candidates when a certain vote is bribed:

1. Bribing a voter in $V_0 \cup V_1^{\neg c}$ makes c gain two relative points with respect to all but one candidate (the candidate approved aside from c after the bribery; with respect to this candidate, the relative gain is one). The reason is that all but one non-distinguished candidates lose one point and c simultaneously wins one point, while the candidate approved aside from c after the bribery does not lose a point by the bribery, but c wins a point.

2. When the briber bribes a voter in $V_2^{\neg c}$, there are two possibilities. Firstly, the briber can make c win one point against $m-2$ other candidates and two points against one candidate, namely if the voter initially approves of c_i and c_j before the bribery and c and c_j after the bribery (where $1 \leq i, j \leq m-1$, $i \neq j$). Secondly, the briber can make c gain two points against two candidates, zero points against one candidate, and one point against all remaining candidates, if any. This holds for the case where the bribed voter initially approves of c_i and c_j before the bribery and c and c_h after the bribery (it holds $1 \leq i, j, h \leq m-1$, $|\{i, j, h\}| = 3$).

3. Bribing a voter in V_1^c makes c win one point against $m-2$ candidates and zero points against one candidate (the one who is approved aside from c after the bribery).

4. Bribing a voter in V_2^c either does not increase c's gain against any other candidate (namely when the bribed voter approves of c and the same c_j before and after the bribery) or c wins and loses one point against one candidate, respectively, and gains no point against the remaining candidates.

Note that the briber bribes only voters from the first two groups. According to our argument with relative gains, the first class of votes should be bribed with highest priority. We point out that in case merely voters from the latter two groups are left to be bribed, c already has a full approval score and is thus a necessary winner. Our algorithm checks the following cases:

- $score_{(C,V)}(c) + \ell \geq n$. In this case, the answer to our problem is YES since the briber can ensure that each voter in the final election $(C, \bar{V})$ approves of c, no matter how the unchanged, partial votes are actually completed.

- $score_{(C,V)}(c) + \ell < n$ and $\ell > |V_0 \cup V_1^{\neg c}|$. In this case, the briber bribes all voters in $V_0 \cup V_1^{\neg c}$ and further $\ell' := \ell - |V_0 \cup V_1^{\neg c}|$ voters in $V_2^{\neg c}$. Before we continue, we point out the following aspects. Firstly, we accept if and only if it is possible to find $|V_2^{\neg c}| - \ell'$ voters in $V_2^{\neg c}$ who are left unchanged by the briber, and each c_j has at most $score_{(C,\bar{V})}(c) = score_{(C,V)}(c) + \ell$ potential points which are assigned to c_j by (1) bribed voters after the bribery, (2) voters in $V_1^c \cup V_2^c$ who surely are not bribed (as c definitely scores), and (3) voters in $V_2^{\neg c}$ being left unchanged by the briber. Moreover, all voters in $V_0 \cup V_1^{\neg c}$ are bribed by the briber (with highest priority). Thus, they do not contribute to any candidate's pscore (although each of these voters approves of c and one c_j after the bribery; these points, however, are assigned by voters from the first group).

  With these preconsiderations in mind, we define the following b-edge matching problem. We are given a multigraph $G = (\mathscr{V}, \mathscr{E})$ with $\mathscr{V} = C \setminus \{c\}$ (each non-distinguished candidate yields

a vertex in G). Each voter in $V_2^{\neg c}$ approving of c_i and c_j ($1 \leq i < j \leq m-1$) yields an edge $\{c_i, c_j\}$ in $\mathscr{E}$. The capacities are $b(c_j) = score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j)$ ($1 \leq j \leq m-1$).

We claim that the briber can make c a necessary winner by bribing ℓ voters if and only if G yields a matching with $|V_2^{\neg c}| - \ell'$ edges (recall that $\ell' = \ell - |V_0 \cup V_1^{\neg c}|$ denotes the number of voters in $V_2^{\neg c}$ who are bribed by the briber, and that the briber bribes all $\ell - \ell' = |V_0 \cup V_1^{\neg c}|$ voters in $V_0 \cup V_1^{\neg c}$ with highest priority) and the following inequality holds:

$$\Big(\sum_{j=1}^{m-1} (score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j))\Big) - 2(|V_2^{\neg c}| - \ell') - \ell \geq 0 \quad (*).$$

($\Rightarrow$): Suppose that the briber can make c a necessary winner by bribing ℓ voters. As c has $score_{(C,\bar{V})}(c) = score_{(C,V)}(c) + \ell$ points in the final election $(C,\bar{V})$, each c_j has $pscore_{(C,\bar{V})}(c_j) = pscore_{(C,V_1^c \cup V_2^c)}(c_j) + \iota(c_j) + \beta(c_j)$ points after the bribery and this number is at most $score_{(C,\bar{V})}(c)$. In this context, $\beta(c_j)$ denotes the number of bribed voters approving of c_j after the bribery. Moreover, we suppose that c_j is approved by $\iota(c_j)$ voters in $V_2^{\neg c}$ who are left unchanged by the briber.

First assume for contradiction that G does not yield any matching of size $|V_2^{\neg c}| - \ell'$ (nor any larger matching). Then selecting any $|V_2^{\neg c}| - \ell'$ edges in $\mathscr{E}$ implicates that it holds $\iota(c_j) > b(c_j)$ for at least one c_j ($1 \leq j \leq m-1$), where $\iota(c_j)$ is the number of edges incident to c_j in our edge collection.[12] This in turn implies $pscore_{(C,\bar{V})}(c_j) = pscore_{(C,V_1^c \cup V_2^c)}(c_j) + \iota(c_j) + \beta(c_j) \geq pscore_{(C,V_1^c \cup V_2^c)}(c_j) + \iota(c_j) > pscore_{(C,V_1^c \cup V_2^c)}(c_j) + b(c_j) = pscore_{(C,V_1^c \cup V_2^c)}(c_j) + score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j) = score_{(C,\bar{V})}(c)$, and thus c_j beats c for at least one extension of the final election, even though the points that bribed voters assign to c_j are not accounted for.

Thus, a feasible matching of cardinality $|V_2^{\neg c}| - \ell'$ exists. It remains to show that the inequality holds. Since $score_{(C,\bar{V})}(c) - pscore_{(C,\bar{V})}(c_j) \geq 0$ holds for each $j \in [m-1]$, it follows that $\sum_{j=1}^{m-1}(score_{(C,\bar{V})}(c) - pscore_{(C,\bar{V})}(c_j)) \geq 0$. This in turn is equivalent to $\sum_{j=1}^{m-1}(score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j) - \iota(c_j) - \beta(c_j)) \geq 0$ which can be rewritten by $(\sum_{j=1}^{m-1}(score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j))) - (\sum_{j=1}^{m-1}\iota(c_j)) - (\sum_{j=1}^{m-1}\beta(c_j)) = (\sum_{j=1}^{m-1}(score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j))) - 2(|V_2^{\neg c}| - \ell') - \ell \geq 0$, and we are done. This condition holds because of $\sum_{j=1}^{m-1}\beta(c_j) = \ell$ (bribed voters assign ℓ approvals to non-distinguished candidates after the bribery) and since $\sum_{j=1}^{m-1}\iota(c_j)$ equals the total number of approvals that non-distinguished candidates receive from unbribed voters in $V_2^{\neg c}$ after the bribery. Observe further that this number is identical to twice the number of voters in $V_2^{\neg c}$ being left unchanged by the briber as each of them approves of two candidates other than c. According to this, the inequality $(*)$ can be rewritten by $(\sum_{j=1}^{m-1}(b(c_j) - \iota(c_j))) - \ell \geq 0$, provided that a matching with $|V_2^{\neg c}| - \ell'$ edges exists and $\iota(c_j)$ edges in this matching are incident to c_j. As a combined result, the existence of a large enough matching means that c is a necessary winner in the final election when we account for all definite points for c after the bribery and all potential points for non-distinguished candidates except for the ℓ approvals

[12] We again use the expression $\iota(c_j)$ since voters in $V_2^{\neg c}$ left unchanged one-to-one correspond to edges in our edge selection.

that bribed voters assign to non-distinguished candidates after the bribery. In case the inequality holds in addition, c stays a necessary winner when these ℓ approvals are taken into account as well.

We finally point out that—in case there are several matchings of maximum cardinality—it does not matter according to which maximum matching the briber leaves unchanged $|V_2^{\neg c}| - \ell'$ voters and bribes the remaining ℓ' voters in $V_2^{\neg c}$ (the inequality does not change when the briber bribes according to another matching).

($\Leftarrow$): Suppose that there is a matching with $|V_2^{\neg c}| - \ell'$ edges and the inequality $(*)$ holds. In case the maximum matching has larger cardinality, we may pick arbitrary $|V_2^{\neg c}| - \ell'$ edges from the maximum matching and obtain a feasible matching of the desired cardinality. We construct a final election $(C, \bar{V})$ for which c is a necessary winner. First of all, c has $score_{(C,\bar{V})}(c) = score_{(C,V)}(c) + \ell$ points after the bribery. It remains to show that each other candidate has at most $score_{(C,\bar{V})}(c)$ potential points. Our constructed final election looks as follows. (1) All voters in $V_2^c \cup V_1^c$ remain unchanged. (2) All $|V_2^{\neg c}| - \ell'$ voters one-to-one corresponding to the same number of edges in the given matching remain unchanged. (3) The briber bribes all ℓ' voters not one-to-one corresponding to edges in the matching and all $\ell - \ell'$ voters in $V_0 \cup V_1^{\neg c}$ (the latter bribes are fixed in advance, the former ones depend on the edges in the matching).

Observe that due to the capacity constraints it holds $pscore_{(C,V_1^c \cup V_2^c)}(c_j) + \iota(c_j) \leq pscore_{(C,V_1^c \cup V_2^c)}(c_j) + b(c_j) = pscore_{(C,V_1^c \cup V_2^c)}(c_j) + score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j) = score_{(C,\bar{V})}(c)$ for each $j \in [m-1]$, that is, the potential score of each c_j is at most as high as the definite final score of c when we consider all potential approvals for c_j in the final election except the approvals that bribed voters assign to c_j after the bribery. Again, $\iota(c_j)$ denotes the number of edges in the matching incident to c_j.

It remains to argue that bribed voters may assign ℓ approvals to non-distinguished candidates and c is still a necessary winner. Since it holds $\iota(c_j) \leq b(c_j)$, candidate c_j may still receive at most $B(c_j) := b(c_j) - \iota(c_j) (\geq 0)$ potential approvals from the bribed voters after the bribery. Aggregating over all candidates c_j, we obtain $(\sum_{j=1}^{m-1} B(c_j)) - \ell = \sum_{j=1}^{m-1} (b(c_j) - \iota(c_j)) - \ell = \sum_{j=1}^{m-1} (score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j) - \iota(c_j)) - \ell = \sum_{j=1}^{m-1} (score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j)) - \sum_{j=1}^{m-1} \iota(c_j) - \ell = \sum_{j=1}^{m-1} (score_{(C,\bar{V})}(c) - pscore_{(C,V_1^c \cup V_2^c)}(c_j)) - 2(|V_2^{\neg c}| - \ell') - \ell$. As our inequality $(*)$ holds, we obtain $(\sum_{j=1}^{m-1} B(c_j)) - \ell \geq 0$. In other words, bribed voters may assign ℓ approvals to non-distinguished candidates and c is (still) a necessary winner when accounting for these ℓ points. These approvals can be greedily allocated:

- The first $\beta(c_1) := \min(B(c_1), \ell)$ bribed voters approve of c and c_1 (w.l.o.g. the lexicographically smallest $\min(B(c_1), \ell)$ voters who are bribed).
- The next $\beta(c_2) := \min(B(c_2), \ell - \beta(c_1))$ bribed voters approve of c and c_2.
- $\vdots$
- The remaining $\beta(c_{m-1}) := \min(B(c_{m-1}), \ell - \beta(c_1) - \beta(c_2) - \ldots - \beta(c_{m-2}))$ bribed voters approve of c and c_{m-1}.

These numbers $\beta(c_j)$—the numbers of additional approvals—are restricted from above by $B(c_j)$ (more approvals would imply that c_j beats c for some completions) and $\ell - \sum_{i=1}^{j-1} \beta(c_i)$ (otherwise, the total number of additional approvals for non-distinguished candidates would exceed ℓ). As $\sum_{j=1}^{m-1} B(c_j) \geq \ell$ and our greedy algorithm assigns as many approvals as possible to each presently regarded c_j, this greedy procedure surely leads to a feasible bribing strategy. Suppose for contradiction that this greedy procedure does not assign ℓ approvals to non-distinguished candidates. Then, for each c_j, it holds $\min(B(c_j), \ell - \sum_{i=1}^{j-1} \beta(c_i)) = B(c_j) < \ell - \sum_{i=1}^{j-1} \beta(c_i) = \ell - \sum_{i=1}^{j-1} B(c_i)$ for each $j \in [m-1]$ (as fewer than ℓ approvals are assigned due to our assumption; otherwise, the sum over all $\beta(c_j)$ is ℓ according to our construction). In particular, it holds $B(c_{m-1}) < \ell - B(c_1) - \ldots - B(c_{m-2})$. This in turn equals $B(c_1) + \ldots + B(c_{m-1}) < \ell$ which is a contradiction to our assumption.

- $\ell \leq |V_0 \cup V_1^{\neg c}|$ and $score_{(C,V)}(c) + \ell < n$. Under these conditions, c cannot necessarily reach a full approval score after the bribery. Moreover, the briber bribes ℓ arbitrary voters in $V_0 \cup V_1^{\neg c}$ and makes each bribed voter approve of c after the bribery. Note that all c_j (temporarily) lose ℓ potential approvals given to them by the bribed voters before the bribery, but again the bribed voters assign ℓ approvals to non-distinguished candidates after the bribery. It follows that the briber can make c a necessary winner by bribing ℓ voters if and only if the following two conditions hold:

$$score_{(C,V)}(c) + \ell \geq pscore_{(C,V)}(c_j) - \ell \quad \forall j \in [m-1]$$

and

$$\ell \leq \sum_{j=1}^{m-1} (score_{(C,V)}(c) + 2\ell - pscore_{(C,V)}(c_j)).$$

The argument of correctness follows similarly to the previous subcase, without arguing with matching. The first condition must hold as otherwise c is beaten by some c_j for some extension even without considering any approvals that c_j might receive from bribed voters after the bribery. The second inequality states that ℓ additional points from the bribed voters may go to the other candidates without c being beaten by any c_j. These approvals can be assigned by a similar greedy algorithm to the one in the previous subcase.

Notice that the first and third subcase are in P because our problem widely reduces to computing the (p)scores (which is possible in $O(mn)$ time), checking inequalities, and applying a greedy algorithm. The second subcase polynomially reduces our problem to computing a maximum b-edge matching and verifying an inequality. ❑

Observe that for $V_0 \cup V_1 = \emptyset$ our problem coincides with bribery under full information in a sense that votes with two or more candidates in their top set assign a unique score to every candidate in C. While the original proof tailored to complete votes works with b-edge cover [119], we provide an alternative proof with b-edge matching.

The following example shall illustrate the matching algorithm in the proof of Theorem 3.19.

Example 3.20. *Let (C,V) be an election with $C = \{c, c_1, c_2, c_3\}$, $V = \{v_1, \ldots, v_7\}$, distinguished candidate c, and bribery limit $\ell = 2$. The underlying partial information model is TTO. V can be partitioned into $V_0 = \{v_1\}$, $V_2^c = \{v_2\}$, and $V_2^{\neg c} = \{v_3, \ldots, v_7\}$. The voters in $V_2 = V_2^{\neg c} \cup V_2^c$ vote as follows (we indicate only the two candidates approved):*

$$v_2 : (c, c_1), \quad v_3 : (c_1, c_2), \quad v_4, v_5 : (c_1, c_3), \quad v_6, v_7 : (c_2, c_3).$$

Figure 3.2 illustrates the matching graph (the one on the left) according to Example 3.20. Based on this instance, we obtain a graph $G = (\mathscr{V}, \mathscr{E})$ with vertices $\mathscr{V} = \{c_1, c_2, c_3\}$, edges $\mathscr{E} = \{e_3, \ldots, e_7\}$ (where e_i is incident to the vertices named after the two candidates approved by voter v_i, $3 \leq i \leq 7$), and capacities $b(c_1) = 2 = score_{(C,V)}(c) + \ell - pscore_{(C,V_1^c \cup V_2^c)}(c_1) = 1 + 2 - 1 = 2$ and $b(c_2) = b(c_3) = 3$ (as they are not potentially approved by any voters who definitely approve of c). Observe that $\ell = 2$ and $\ell' = 1$ (one bribe is reserved for the voter in V_0, the other bribe concerns the set $V_2^{\neg c}$).

In our example, the maximum matching has a cardinality of four (e.g., all edges but e_5). Nevertheless, there is no successful bribery as it holds $(\sum_{j=1}^{3}(score_{(C,V)}(c) + \ell - pscore_{(C,V_1^c \cup V_2^c)}(c_j))) - 2(|V_2^{\neg c}| - \ell') - \ell = (2+3+3) - 2 \cdot 4 - 2 = -2 \not\geq 0$. Or, in other words, leaving all voters according to the matching unchanged implicates that c, c_1, c_2, and c_3 have a (p)score of 3 each, not accounting for the approvals for c_1, c_2, and c_3 assigned by the bribed voters. However, as soon as one bribed voter approves of some c_j, c is beaten by c_j.

The right graph corresponds to a variation of Example 3.20 with the difference that voter v_5 does not occur in V. Observe that there is now a maximum matching consisting of all edges in $\mathscr{E}$. Now our instance is a YES-instance: the matching (cardinality 4) is large enough as we require a matching with at least $|V_2^{\neg c}| - \ell' = 4 - 1 = 3$ edges. Moreover, our inequality holds: $2 + 3 + 3 - 2 \cdot 3 - 2 = 0 \geq 0$. Suppose that the briber leaves unchanged the voters one-to-one corresponding to e_3, e_4, and e_6 in the matching and bribes v_7 (and v_1). Then the "preliminary scores" of c_1, c_2, and c_3 are 3, 2, and 2, respectively (by "preliminary", we mean the final scores of the c_j without accounting for $\ell = 2$ approvals the bribed voters still have to assign to them). By making the bribed voters approve of c twice and c_2 and c_3 once each, we obtain a final (p)score of 3 for all candidates in C and c is a necessary winner. Note that another bribery strategy (e.g., according to the matching e_3, e_6, and e_7) yields a YES-instance as well since then the bribed voters may approve of c_1 and c_2, aside from c, and again all candidates have three points.

We obtain yet another polynomial-time result for the 1TOS structure.

Theorem 3.21. 2-APPROVAL-1TOS-NECESSARY BRIBERY *is in* P.

Proof. Let $\tilde{C} \subseteq C$ be the candidate subset completely ranked by each voter. First of all, if $c \notin \tilde{C}$, we can regard an equivalent instance with $\tilde{C}' := \tilde{C} \cup \{c\}$ and all voters rank c last among the candidates in $\tilde{C}'$. Hence, we let $c \in \tilde{C}$. Our algorithm distinguishes the following cases:

- $\tilde{C} = C$. This case corresponds to bribery under full information which is in P [119].
- $\tilde{C} = \{c\}$. In this case, each voter potentially approves of all non-distinguished candidates and c is never approved in his worst-case. Thus, we have $pscore(d) = n$ for every candidate $d \neq c$ and $score(c) = 0$. We can solve this subcase by instead regarding a TTO instance with $V = V_0$ and apply the algorithm in the proof of Theorem 3.19.

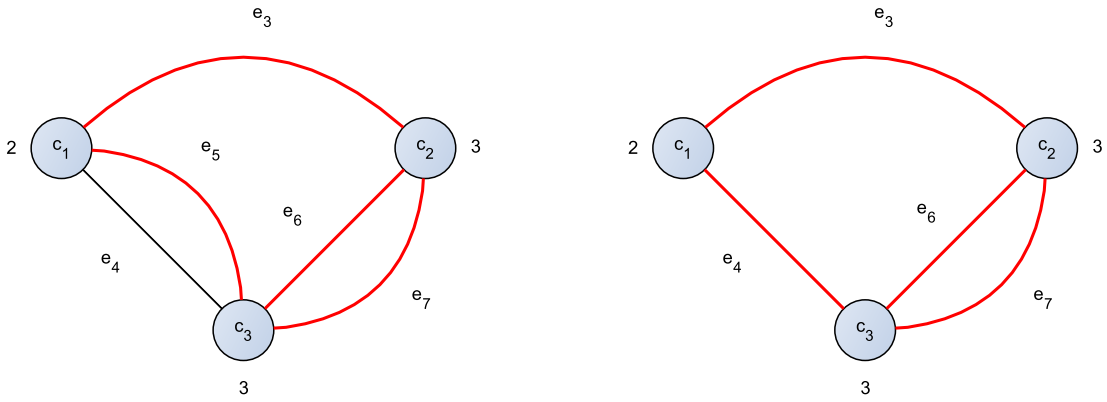

Figure 3.2: Two examples of the matching graphs according to Example 3.20. The left graph corresponds to the original example, while the graph on the right represents the variation of the original example. The edges in a maximum matching are marked red. The vertex capacities are written next to the vertices.

- $c \in \tilde{C}$, $C \setminus \tilde{C} = \{p\}$. In this case, we have $score(c) = s_1(c)$, where $s_i(c)$ $(i = 1,2)$ is defined as the number of voters ranking c on position i in subelection $(\tilde{C},V)$. Moreover, we have $pscore(c') = score_{(\tilde{C},V)}(c') = s_1(c') + s_2(c')$ for each $c' \in \tilde{C} \setminus \{c\}$ and $pscore(p) = n$. In other words, each voter potentially approves of p. In addition, each voter definitely approves of his favorite candidate in $\tilde{C}$ and potentially approves of his second most preferred candidate in $\tilde{C}$ unless this candidate is c (as unsure approvals do not count for c in c's worst-case).

  First, if $score(c) + \ell < n - \ell$, we reject since p has $n - \ell$ or more potential points in the final election, while c has merely $score(c) + \ell = s_1(c) + \ell$ definite points after the bribery.

  If $score(c) + \ell \geq n$, we accept as the briber can make c achieve a full score for every extension.

  It remains to argue for $n - \ell \leq score(c) + \ell < n$. In this case, the briber cannot make c reach a full approval score for at least one extension. Nonetheless, c can reach p and all non-distinguished candidates in $\tilde{C}$ when we do not account for the ℓ approvals that bribed voters assign to candidates other than c after the bribery. The reason is as follows. The distinguished candidate c is definitely approved by $score(c) + \ell$ voters after the bribery. Each non-distinguished candidate is potentially approved by at most $n - \ell$ unbribed voters and possibly by some bribed voters. Due to our assumption, we have $score(c) + \ell \geq n - \ell$.

  We will show that the briber should bribe as many (arbitrary) voters v as possible with $pos_{\tilde{C}}(c,v) > 2$ and as few (arbitrary) voters v as possible with $pos_{\tilde{C}}(c,v) = 2$. Moreover, the briber does not bribe any voters v with $pos_{\tilde{C}}(c,v) = 1$ as c definitely scores in these votes.

  Observe that the briber leaves unchanged $n - s_1(c) - \ell$ voters v with $pos_{\tilde{C}}(c,v) \geq 2$ and bribes the ℓ remaining of these voters. Assume that among these $n - s_1(c) - \ell$ voters, $\lambda(c') \in \{0,\ldots,n - s_1(c) - \ell\}$ voters potentially approve of $c' \in \tilde{C} \setminus \{c\}$. [13] (Note that all voters left unchanged potentially approve of p.) Then the bribed voters may still assign

[13] Actually, $\lambda(c')$ is bounded from above by $n - s_1(c) - \ell$ and by the number of voters v with $pos_{\tilde{C}}(c,v) > 1$ potentially approving of c'.

$B(c') := s_1(c) + \ell - pscore_{(C,V^c)}(c') - \lambda(c')(\geq 0)$ approvals to c' after the bribery (where V^c denotes the set of voters definitely approving of c), and the final pscore of c' does not exceed the final definite score of c. Moreover, p may receive further $B(p) := s_1(c) + 2\ell - n$ approvals from the bribed voters (this number holds regardless of the ℓ voters bribed by the briber).

We accept for this subcase if and only if the sum over all $B(d)$, $d \in C \setminus \{c\}$, is at least ℓ. Otherwise, accounting for ℓ further approvals for non-distinguished candidates implicate that for each extension some non-distinguished candidate beats c.

By aggregating over all $B(d)$, $d \in C \setminus \{c\}$, we obtain

$$\sum_{d \in C \setminus \{c\}} B(d) = \Big(\sum_{c' \in \tilde{C} \setminus \{c\}} (s_1(c) + \ell - pscore_{(C,V^c)}(c') - \lambda(c')) \Big) + (s_1(c) + 2\ell - n)$$

$$= \Big(\sum_{c' \in \tilde{C} \setminus \{c\}} (s_1(c) + \ell) \Big) - \Big(\sum_{c' \in \tilde{C} \setminus \{c\}} pscore_{(C,V^c)}(c') \Big) - \Big(\sum_{c' \in \tilde{C} \setminus \{c\}} \lambda(c') \Big) + (s_1(c) + 2\ell - n).$$

As c is definitely approved by $s_1(c)$ voters, it follows that $\sum_{c' \in \tilde{C} \setminus \{c\}} pscore_{(C,V^c)}(c') = s_1(c)$. Now let us assume that β voters v with $pos_{\tilde{C}}(c,v) \geq 3$ and α voters v with $pos_{\tilde{C}}(c,v) = 2$ are left unchanged by the briber. Notice that it holds $\alpha + \beta = n - s_1(c) - \ell$. Since each voter ranking c on second position, restricted to $\tilde{C}$, potentially approves of only one candidate in $\tilde{C} \setminus \{c\}$ and since all voters v with $pos_{\tilde{C}}(c,v) \geq 3$ potentially approve of two candidates in $\tilde{C} \setminus \{c\}$, it holds $\sum_{c' \in \tilde{C} \setminus \{c\}} \lambda(c') = 2\beta + \alpha$ and we obtain

$$\sum_{d \in C \setminus \{c\}} B(d) = (|\tilde{C}| - 1) \cdot (s_1(c) + \ell) - s_1(c) - 2\beta - \alpha + (s_1(c) + 2\ell - n).$$

This number is maximized when β is as small as possible and α is as large as possible (that is, as many voters v as possible with $pos_{\tilde{C}}(c,v) = 2$ remain unchanged by the briber). Notice that $\hat{\alpha} = \min(s_2(c), n - s_1(c) - \ell)$ and $\hat{\beta} = n - s_1(c) - \ell - \hat{\alpha}$ is the combination maximizing the sum over all $B(d)$. According to the previous reasoning, it follows that the briber can make c a necessary winner if and only if

$$(|\tilde{C}| - 1) \cdot (s_1(c) + \ell) - s_1(c) - 2\hat{\beta} - \hat{\alpha} + (s_1(c) + 2\ell - n) \geq \ell.$$

Then, and only then, c is a necessary winner of the final election after ℓ bribed voters have assigned ℓ approvals to non-distinguished candidates (recall from the reasoning above that c is definitely a necessary winner in the final election when we do not consider these ℓ approvals). Observe that this inequality holds regardless of which voters are actually bribed according to the "bribing strategy" $(\hat{\alpha}, \hat{\beta})$.

- $2 \leq |\tilde{C}| \leq |C| - 2$. In this case, there are at least two non-distinguished candidates p with $p \notin \tilde{C}$. As there are extensions where two candidates in $C \setminus \tilde{C}$ are approved by each voter, we

have $score(c) = 0$, $pscore(p) = n$ ($p \in C \setminus \tilde{C}$), and $pscore(c') = score_{(\tilde{C},V)}(c') - |\{v \in V : c \succ_v c' \succ_v d \; \forall d \in \tilde{C} \setminus \{c,c'\}\}|$ for $c' \in \tilde{C} \setminus \{c\}$. [14]

First of all, if $\ell < \frac{n}{2}$, our instance is a NO-instance as the candidates in $C \setminus \tilde{C}$ have $n - \ell$ or more potential points in the final election which is higher than c's definite final score ℓ.

If $\ell \geq \frac{n}{2}$ and $\ell \geq n - s_1(c)$, our instance is a YES-instance. The briber bribes all voters v with $pos_{\tilde{C}}(c,v) \geq 2$ and $\ell - n + s_1(c)$ voters v with $pos_{\tilde{C}}(c,v) = 1$. By making all bribed voters approve of c and w.l.o.g. the same $c' \in \tilde{C} \setminus \{c\}$, we obtain $score_{(C,\bar{V})}(c) = pscore_{(C,\bar{V})}(c') = \ell$, $pscore_{(C,\bar{V})}(c'') = 0$ ($c'' \in \tilde{C} \setminus \{c,c'\}$; only relevant when $|\tilde{C}| > 2$), and $pscore_{(C,\bar{V})}(p) = n - \ell$. Observe that no non-distinguished candidate has a higher score than c for any extension.

Hence, it remains to argue for $\frac{n}{2} \leq \ell < n - s_1(c)$. In this subcase, the briber does not bribe any voters with favorite candidate c restricted to the candidates in $\tilde{C}$ as these voters only contribute to the pscore values of the candidates in $C \setminus \tilde{C}$, whereas all other voters potentially approve of at least one non-distinguished candidate in $\tilde{C}$ in addition.

Since the briber can bribe at least half of all voters and ensure a definite score of ℓ for c, all non-distinguished candidates have at most $n - \ell \leq \ell = score_{(C,\bar{V})}(c)$ potential points in the final election each when we consider all approvals meant for them except for those ℓ approvals that bribed voters assign to them after the bribery.

Analogously to the case $C \setminus \tilde{C} = \{p\}$, $p \neq c$, it follows that the briber should bribe as many voters v as possible with $pos_{\tilde{C}}(c,v) \geq 3$ and leave unchanged as many voters v as possible with $pos_{\tilde{C}}(c,v) = 2$. According to our assumptions, the briber does not bribe any voters v with $pos_{\tilde{C}}(c,v) = 1$. In contrast to the other case, we have to consider the following aspects. Firstly, we have $score(c) = 0$ and therefore $pscore_{(C,V^c)}(c') = 0$ for each $c' \in \tilde{C} \setminus \{c\}$. Secondly, $n - s_1(c) - \ell$ voters not ranking c first in subelection $(\tilde{C},V)$ are ignored by the briber and the remaining of these voters are bribed. In opposition to the previous case, the $s_1(c)$ voters ranking c first, restricted to $\tilde{C}$, do not increase the (p)score of any candidate in $\tilde{C}$, but again these voters are not bribed in this subcase. Thirdly, the bribed voters may assign a total of $|C \setminus \tilde{C}| \cdot (2\ell - n)$ approvals to candidates in $C \setminus \tilde{C}$ and c necessarily beats or ties with all of them.

Again, we assume that the briber ignores $\hat{\alpha} := \min(n - s_1(c) - \ell, s_2(c))$ voters v with $pos_{\tilde{C}}(c,v) = 2$ and $\hat{\beta} := n - s_1(c) - \ell - \hat{\alpha}$ voters v with $pos_{\tilde{C}}(c,v) \geq 3$ and bribes the remaining ℓ voters v with $pos_{\tilde{C}}(c,v) > 1$. We define $B(d)$, $d \in C \setminus \{c\}$, analogously to the previous case and set $B(c') = \ell - \lambda(c')$ for $c' \in \tilde{C} \setminus \{c\}$ (where $\lambda(c') \in \{0, \ldots, \min(pscore(c'), n - s_1(c) - \ell)\}$ denotes the number of voters ignored by the briber who (1) potentially approve of c' and (2) do not rank c first among the candidates in $\tilde{C}$) and $B(p) = 2\ell - n$ for each $p \in C \setminus \tilde{C}$. Likewise, we accept if and only if $\sum_{d \in C \setminus \{c\}} B(d) \geq \ell$, that is, our instance is a YES-instance if and only if

$$\ell \leq (|C \setminus \tilde{C}| \cdot (2\ell - n)) + ((|\tilde{C}| - 1)\ell - 2\hat{\beta} - 1\hat{\alpha}).$$

[14] c' receives all potential points that c' would gain in subelection $(\tilde{C},V)$, except for those voters ranking c first and c' second restricted to candidates in $\tilde{C}$: although c' is approved by these voters for some extensions, c''s approval does not count in c's worst-case. The reason is as follows. Since c' can never achieve a higher score than c in these votes for any extension, we assume that c and c' are ranked outside an approval position each, behind all candidates in $C \setminus \tilde{C}$ who all are potentially approved by these voters. If $|C \setminus \tilde{C}| = 2$, we obtain full information for these votes in a sense that both "outside" candidates get an approval score of one in these votes and all candidates in $\tilde{C}$ are supposed to be ranked behind.

It follows that our decision problem belongs to P as each subcase either directly reduces to bribery under full information (which is known to be easy due to [119]), can be solved via the algorithm in the proof of Theorem 3.19 tailored to the TTO model, or we just have to check some polynomially bounded inequalities. ❑

Although bribery under full information is easy both for Plurality and 2-Approval, we could observe that six models of partial information make the complexity jump from P to NP-completeness, whereas three models preserve the polynomial-time decidability.

By contrast, we know that for $k \geq 3$ bribery in k-Approval is NP-complete under full information [119]. Consequently, according to Theorem 3.15, necessary bribery is NP-complete under all nine structures of partial information as well.

Corollary 3.22. *k-APPROVAL-X-NECESSARY BRIBERY is NP-complete for each $k \geq 3$ and each $X \in PIM$.*

Now let us turn to the Veto rule. Opposed to Plurality, the complexity of bribery under Veto does not increase for any structure of partial information compared to full information.

Theorem 3.23. *VETO-X-NECESSARY BRIBERY is in P for every model $X \in PIM$.*

Proof. It suffices to regard the two most general models, FP and PC. Basically, there are three different kinds of votes v:

1. c is potentially vetoed by v, regardless of whether v potentially vetoes other candidates as well.
2. Voter v certainly vetoes c_j ($j \in \{1,\ldots,m-1\}$).
3. Voter v potentially vetoes the pairwise different non-distinguished candidates $c_{j_1},\ldots,c_{j_r}$ ($2 \leq r \leq m-1$).

We assume that all potential vetoes count for c or, in other words, all voters from the first group veto c for some extensions. Hence, c has $pvetoes(c)$ vetoes in the original election. Moreover, we suppose that each c_j is only vetoed by voters from the second group. We may argue like this as we want to make c beat or tie with c_j in extensions where all potential vetoes count for c, but only the definite vetoes for c_j are considered. As we wish to ensure this for all c_j, we may assume that voters from the third group do not increase anyone's veto number. Therefore, we count the number of definite vetoes for each c_j, denoted by $vetoes(c_j)$ ($1 \leq j \leq m-1$).

A rational briber bribes as many voters as possible who potentially veto c. Moreover, every bribed voter does not veto c after the bribery. Similarly to the proof under full information [79], a greedy algorithm can be applied: each bribed voter vetoes the non-distinguished candidate with the currently lowest veto number. We distinguish the following cases.

If $\ell \geq pvetoes(c)$, we accept because the briber can make c reach zero vetoes for every extension which is sufficient for c to be a necessary winner in the co-winner model.

For $\ell < pvetoes(c)$, the briber bribes arbitrary ℓ voters potentially giving their veto to c and makes them veto other candidates after the bribery. Since c has $pvetoes(c) - \ell$ vetoes in the final election, candidate c_j requires $\delta(c_j) := \max(0, pvetoes(c) - \ell - vetoes(c_j))$ additional vetoes from

the bribed voters in order to not beat c. It remains to check whether ℓ additional vetoes from bribed voters (after the bribery) suffice to give $\delta(c_j)$ new vetoes to each c_j. It follows that c is a necessary winner after the bribery of ℓ voters if and only if $\ell \geq \sum_{j=1}^{m-1} \delta(c_j)$ holds. As this condition can be checked in polynomial time and the preprocessings from above are easy as well, our overall problem is in P. ❑

Before considering the complexity of necessary bribery in 2-Veto, we first settle the following auxiliary result for k-Veto.

Lemma 3.24. *Let $(\delta(c_1),\ldots,\delta(c_{m-1})) \in \mathbb{N}_0^{m-1}$ (one can interpret $\delta(c_j)$ as the additional number of vetoes c_j still requires from the bribed voters). Then, given k-Veto ($k < m$), the following condition holds:*

ℓ voters can feasibly assign $\delta(c_j)$ vetoes to c_j (for each $j = 1,\ldots,m-1$) if and only if it holds

$$\delta(c_j) \leq \ell \quad (1 \leq j \leq m-1) \quad \text{and} \quad \sum_{j=1}^{m-1} \delta(c_j) \leq k\ell.$$

Lemma 3.24 provides us a formalization of the intuitive result that ℓ bribed voters can greedily assign at most ℓ voters to each c_j such that their total number of required vetoes does not exceed the total number of vetoes that ℓ bribed voters assign, namely $k \cdot \ell$. For better readability, we defer the proof to the Appendix and instead give a short example how these vetoes are greedily assigned.

Example 3.25. *Let $C = \{c, c_1, \ldots, c_5\}$, $k = 3$, and $|V| > \ell = 4$ (where $|V|$ is not specified). The needs of additional vetoes are as follows:*

$$\delta(c_1) = 3, \quad \delta(c_2) = 4, \quad \delta(c_3) = 3, \quad \delta(c_4) = 0, \quad \delta(c_5) = 1.$$

According to Lemma 3.24, the four voters can distribute the $k\ell = 3 \cdot 4 = 12$ vetoes to the five non-distinguished candidates such that each of them gets at least $\delta(c_j) \leq 4$ vetoes and their total required number of additional vetoes is at most 12—the total veto number assigned by bribed voters: it holds $\delta(c_j) \leq 4$ for each $j \in \{1,\ldots,5\}$ and $\delta(c_1) + \ldots + \delta(c_5) \leq 12$. Let A_j ($1 \leq j \leq 5$) denote the (4×3)-matrix after j candidates have been assigned their vetoes. In this matrix, a_{ir} contains the candidate whom the ith bribed voter ranks on his rth best veto position (note that according to this, a_{i3} contains voter i's candidate ranked last, whereas a_{i1} represents i's third to least preferred alternative). Our algorithm defined in the proof in the Appendix assigns the vetoes as follows:

$$A_1 = \begin{pmatrix} c_1 & * & * \\ c_1 & * & * \\ c_1 & * & * \\ * & * & * \end{pmatrix}, \quad A_2 = \begin{pmatrix} c_1 & c_2 & * \\ c_1 & c_2 & * \\ c_1 & c_2 & * \\ c_2 & * & * \end{pmatrix}, \quad A_3 = \begin{pmatrix} c_1 & c_2 & c_3 \\ c_1 & c_2 & c_3 \\ c_1 & c_2 & * \\ c_2 & c_3 & * \end{pmatrix},$$

$$A_4 = \begin{pmatrix} c_1 & c_2 & c_3 \\ c_1 & c_2 & c_3 \\ c_1 & c_2 & * \\ c_2 & c_3 & * \end{pmatrix}, \quad A_5 = \begin{pmatrix} c_1 & c_2 & c_3 \\ c_1 & c_2 & c_3 \\ c_1 & c_2 & c_5 \\ c_2 & c_3 & * \end{pmatrix}.$$

Loosely speaking, our algorithm starts in the leftmost column in the upper left field of the matrix and moves down as long as possible in the current column. As soon as the present column is entirely

filled, the algorithm moves to the top of the next column on the right and again moves down as long as possible. After five iterations, each c_j has been assigned to $\delta(c_j)$ bribed voters, the algorithm successively fills exactly $\sum_{j=1}^{m-1} \delta(c_j) = 11$ entries of the matrix with candidates, not leaving out any field "on its way", and no candidate is vetoed twice or thrice by the same voter. The reason is as follows. Either a candidate fits in one column of the matrix (such as c_1 or c_5) or there is a column break without any overlaps. E.g., the fourth voter gives his first veto to c_2 and the first three voters assign their second veto to c_2 each.

In our example, there is one veto not yet assigned—the third veto assigned by the fourth voter. As voter v_4 already vetoes c_2 and c_3, we set $a_{43} = c_1$ (w.l.o.g. the lexicographically smallest candidate not yet vetoed by voter 4). This leads to the following final veto distribution:

$$A = \begin{pmatrix} c_1 & c_2 & c_3 \\ c_1 & c_2 & c_3 \\ c_1 & c_2 & c_5 \\ c_2 & c_3 & c_1 \end{pmatrix}.$$

Observe that this algorithm is very intuitive and based on a simple idea. We point out that a similar procedure might already exist in a very different setting. Just in case, we have provided this simple and yet non-trivial result.

Our next result states that necessary bribery in 2-Veto is easy for all models in PIM.

Theorem 3.26. 2-VETO-X-NECESSARY BRIBERY *is in* P *for every model* $X \in$ *PIM.*

Proof. It suffices to show our result for FP and PC. All other results follow. Regardless of which model we study, we count all potential vetoes for c and all definite vetoes for each c_j. We point out that this is entirely correct for the FP structure and almost correct for partial orders. For PC, it may occur that c is possibly but not definitely vetoed by a voter v, and v votes $c \succ_v c_j$ for some c_j. Then, in c's worst-case, c is fixed to position $m-1$ and c_j to position m—regardless of whether c_j's veto is definite or not. Observe that these extensions hurt c the most as c performs as badly as possible compared to all candidates c_i ($i \neq j$) and not better than c_j, for c and c_j get the same score from v (v can be extended such that c is not vetoed and c_j is vetoed, but not the other way round; thus, c_j can never do better than c in v).

In both models, we obtain (at most) the following kinds of votes, specifying the two veto positions (we have $i \neq j$, $1 \leq i, j \leq m-1$):

$$(c, c_j), \quad (c, *), \quad (c_i, c_j), \quad (c_j, *), \quad (*, *).$$

$*$ stands for an uncertain veto that surely does not go to c. Now the following three observations help us for the remainder of the proof. Firstly, a rational briber bribes only voters potentially vetoing c because this increases the relative gain against each other candidate compared to the latter three types of votes. Secondly, bribed voters do not veto c after the bribery. Finally, voters of the type $(c, *)$ should be bribed rather than voters of the kind (c, c_j). Bribing a voter of the second type, only c loses a veto, whereas bribing the first kind of voters also decreases the veto number of a non-distinguished candidate.

For our purposes, we let $pvetoes(c) =: v_0(c) + v_1(c)$ where $v_0(c)$ counts the number of voters potentially vetoing c and for whom the other vetoed candidate is not uniquely known, while $v_1(c)$

denotes the number of voters definitely vetoing c_j and potentially vetoing c. [15] Our algorithm distinguishes the following cases:

- $\ell \leq v_0(c)$ and $\ell < pvetoes(c)$. Then the briber bribes ℓ arbitrary voters with c being the only definite veto candidate. As c has $pvetoes(c) - \ell$ vetoes after the bribery, the briber succeeds when he manages to give $\delta(c_j) := \max(0, pvetoes(c) - \ell - vetoes(c_j))$ vetoes to c_j ($j \in [m-1]$). $\delta(c_j)$ is the smallest number of vetoes that c_j additionally needs from bribed voters to not beat c in the final election for any extension. It follows from Lemma 3.24 that the briber can make c a necessary winner after the bribery by bribing ℓ voters of the type $(c, *)$ if and only if

$$(\sum_{j=1}^{m-1} \delta(c_j)) \leq 2\ell \quad \wedge \quad \ell \geq \max_{j \in [m-1]} \delta(c_j)$$

  holds. Then—and only then—2ℓ vetoes from bribed voters after the bribery suffice that each c_j can get at least $\delta(c_j)$ additional vetoes in order to not beat c in the new election, and that no candidate requires more than ℓ vetoes in total.

- $\ell > v_0(c) \ \wedge \ \ell < pvetoes(c)$. In this case, the briber bribes all $v_0(c)$ voters with c as the only definite veto candidate and $\ell - v_0(c)$ further voters with two unique veto candidates (and c among them). Observe that—in order to keep the minimum veto numbers for the c_j as high as possible—the briber should bribe a voter vetoing c and a candidate c_j with the currently highest veto number. After updating the veto numbers in each step and performing $\ell - v_0(c)$ such bribes, the briber greedily distributes 2ℓ vetoes according to the distribution scheme in the previous case, that is, at most 2ℓ additional vetoes are given to non-distinguished candidates c_j and none of them gets more than ℓ vetoes. It remains to validate whether c is a necessary winner or not after applying this strategy, by comparing c's number of potential vetoes with the numbers of definite vetoes of all other candidates.

- $pvetoes(c) \leq \ell$. In this case, the answer to our problem is YES since the briber can take away all potential vetoes from c and make all these voters veto arbitrary other candidates. Hence, c has zero vetoes after the bribery for each completion and this is sufficient to be a winner in the co-winner model.

❑

As we could observe in the proofs of Theorem 3.23 and 3.26, our instance is a YES-instance under the co-winner model whenever the briber can ensure that c has zero potential vetoes after the bribery. For the unique-winner model, this does not hold anymore as the briber must guarantee one or more definite vetoes for every c_j. Our next result shows that 3-VETO-BRIBERY, that is, bribery

[15] To avoid confusion, we point out that c's veto may actually be uncertain, but unsure vetoes are regarded as definite vetoes in the worst case of c. Also remind that—under partial orders—both the vetoes for c and c_j may be unsure and the voter prefers c to c_j, but both candidates are fixed on the two veto positions in c's worst-case. Hence, the word "definitely" refers to our election after implementing our preconsiderations from above.

under full information, is hard under the unique-winner model even though no voter vetoes c in the initial election. [16] In fact, the briber bribes only voters not vetoing c.

Theorem 3.27. 3-VETO-BRIBERY *is* NP*-complete under the unique-winner model.*

Proof. We are given a 3DM instance (X,Y,Z,E), with $|X| = |Y| = |Z| = n$ and triples $E = \{e_1,\ldots,e_m\} \subseteq X \times Y \times Z$. Moreover, we let $m \geq n$ because otherwise a three-dimensional matching is impossible. Based on this 3DM instance, we construct the following 3-VETO-BRIBERY instance.

Let $C = \{c\} \dot\cup X \dot\cup Y \dot\cup Z \dot\cup F$, with $F = \{f_1,\ldots,f_{3m-3n}\}$, be the set of candidates including $3m+1$ candidates and c the designated candidate. We further let V be the set of voters containing m voters v_i, $1 \leq i \leq m$, where each v_i vetoes exactly the candidates (x^i,y^i,z^i) according to the triple $e_i \in E$ (with $x^i \in X$, $y^i \in Y$, and $z^i \in Z$ for each i, $1 \leq i \leq m$). Let the bribery limit be $\ell = m-n$. Finally, $l(a)$ denotes the number of triples in E containing $a \in X \cup Y \cup Z$.

Note that $\mathit{vetoes}(c) = 0$, $\mathit{vetoes}(f_j) = 0$, for each j with $1 \leq j \leq 3m-3n$, and $\mathit{vetoes}(a) = l(a)$ for each $a \in X \cup Y \cup Z$.

We claim that there is a three-dimensional matching of size n if and only if c can be made the unique 3-Veto winner of the election by bribing at most $m-n$ voters.

($\Rightarrow$): Assume there is a three-dimensional matching of size n. Let the briber bribe the $m-n$ voters not corresponding to the matching and veto in each vote three different candidates from F.

After the bribery, we obtain the following veto numbers: $\mathit{vetoes}_{(C,\bar V)}(c) = 0$ and $\mathit{vetoes}_{(C,\bar V)}(d) = 1$ for each $d \in C \setminus \{c\}$; thus, c is the unique 3-Veto winner of the resulting election.

($\Leftarrow$): Assume that we can make c the unique 3-Veto winner of the election by changing at most $m-n$ votes. This implies that $3m-3n$ candidates in F must receive a veto in the bribed votes and this is only possible if each bribed voter vetoes three different candidates in F such that none of them is vetoed at least twice and all candidates in F are vetoed exactly once. On the other hand, the remaining n voters must veto at least once each candidate in X, Y, and Z; otherwise, there would be a candidate other than c with zero vetoes, hence tying with c for first place. Since there are $3n$ candidates in the sets X, Y, and Z and n voters can veto exactly $3n$ candidates, the candidate triples vetoed by these n voters have to correspond to a three-dimensional matching. ❑

Theorem 3.27 and Theorem 3.15 imply that necessary bribery in 3-Veto is hard for all nine models in PIM under the unique-winner model.

Corollary 3.28. 3-VETO-X-NECESSARY BRIBERY *is* NP*-complete for every model* $X \in$ *PIM under the unique-winner model.*

Our next theorem states that bribery in 3-Veto is easy for all models in PIM under the co-winner model.

Theorem 3.29. *Under the co-winner model,* 3-VETO-X-NECESSARY BRIBERY *is in* P *for all partial information models in PIM.*

Proof. We prove our theorem only for $X \in \{\mathrm{FP},\mathrm{PC}\}$. All other models inherit the polynomial time upper bound from one of these two models. In both models, we count all $\mathit{pvetoes}_{(C,V)}(c)$

[16] This proof (more precisely, the same proof but reducing from R3DM) was submitted to IJCAI 2018 in the context of lot-based voting. The result itself was also presented at my talk at the Doctoral Consortium (ADT 2017 in Luxembourg).

potential vetoes for c, whereas we consider only definite vetoes for the c_j. Under the PC model, this is barely correct as voters possibly but not definitely vetoing c, and preferring c to c_j, first must be rewritten by fixing c on a veto position and c_j behind c. Note that there may be up to two different candidates other than c who are fixed on a veto position according to this, regardless of whether or not their vetoes are uncertain in the original election.

After applying this preprocessing for each vote, we calculate $pvetoes_{(C,V)}(c)$ and $vetoes_{(C,V)}(c_j)$ $(1 \leq j \leq m-1)$ in both models. We partition V into voters in $V^{\neg c}$ definitely not vetoing c and voter group V^c each of whom potentially vetoes c. In contrast to the standard model of full information, the following kinds of votes may occur, with the vetoed candidates being specified (we let $1 \leq h,i,j \leq m-1$, and $|\{h,i,j\}| = 3$):

$$(c_h, c_i, c_j), \quad (c_h, c_i, *), \quad (c_h, *, *), \quad (*, *, *),$$

$$(c, c_h, c_i), \quad (c, c_h, *), \quad (c, *, *).$$

$*$ denotes a veto position for which the vetoed candidate is not uniquely known (those vetoes surely do not go to c as all potential vetoes count for c). The upper row contains only voters from $V^{\neg c}$, each voter according to the lower line belongs to V^c.

First regard the case $|V^c| = pvetoes_{(C,V)}(c) \leq \ell$. We accept because the briber reaches that c has no potential veto at all in the final election, no matter how the votes are completed. The briber achieves this just via bribing all voters in V^c and making all of them veto three arbitrary non-distinguished candidates.

It remains to argue for $\ell < |V^c|$. After the bribery, c has $pvetoes_{(C,V)}(c) - \ell > 0$ potential vetoes. To solve our problem, we define the following generalized b-edge cover problem.

We are given an undirected multigraph $G = (\mathscr{V}, \mathscr{E})$ with vertexes $\mathscr{V} := (C \setminus \{c\}) \dot{\cup} \{b, *, **\}$ and edges $\mathscr{E}$ defined as follows.

- Each voter vetoing c, c_i, and c_j yields an edge between c_i and c_j $(1 \leq i < j \leq m-1)$.
- For each voter vetoing c and c_j $(j \in [m-1])$ and for whom the third vetoed candidate is unknown, there is an edge between c_j and $*$.
- For each voter vetoing c and for whom the other two vetoed candidates are not uniquely determined, there is an edge between $*$ and $**$.
- For each $j \in [m-1]$, there are ℓ edges between b and c_j.

The lower capacities are $b_l(c_j) = \max(0, pvetoes_{(C,V)}(c) - \ell - vetoes_{(C,V^{\neg c})}(c_j))$ for $j \in [m-1]$, $b_l(*) = b_l(**) = 0$, and $b_l(b) = 3\ell$. Moreover, we have $b_u(b) = 3\ell$. All remaining upper capacities are unlimited. Note that each unbribed voter in V^c corresponds to one edge, whereas each bribed voter yields three edges incident to b. The fixed capacity of b ensures that b is covered *exactly* 3ℓ times, that is, bribed voters assign 3ℓ vetoes to non-distinguished candidates and at most ℓ vetoes to each c_j.

We claim that the briber can make c a necessary winner by bribing (w.l.o.g.) exactly ℓ voters if and only if there is a minimum edge cover with at most $|V^c| + 2\ell = pvetoes_{(C,V)}(c) + 2\ell$ edges.

($\Rightarrow$): Assume that a successful bribery exists. Then every c_j has at least $pvetoes_{(C,V)}(c) - \ell$ vetoes in the final election $(C, \bar{V})$. Let $\alpha(c_j)$ be the number of voters in V^c who are not changed by the briber and who veto c_j. Let further $\beta(c_j)$ be the number of bribed voters vetoing c_j after the bribery. Since c is a necessary winner in the final election, it holds $vetoes_{(C,\bar{V})}(c_j) = vetoes_{(C,V^{\neg c})}(c_j) + \alpha(c_j) + \beta(c_j) \geq pvetoes_{(C,\bar{V})}(c) = pvetoes_{(C,V)}(c) - \ell$. We construct the following edge cover satisfying our desired properties:

- Each voter in V^c vetoing c, c_j, and c_i (c, c_j, and the third veto candidate is not uniquely known) and not being bribed yields an edge $\{c_i, c_j\}$ (an edge $\{c_j, *\}$) in the cover. If the voter potentially vetoes c and no non-distinguished candidate is definitely vetoed, we add an edge $\{*, **\}$ to the cover.

- For each bribed voter v, there are three edges $\{b, c_1^v\}$, $\{b, c_2^v\}$, and $\{b, c_3^v\}$ in the cover (where c_h^v, $1 \leq h \leq 3$, are the three pairwise different non-distinguished candidates vetoed by v after the bribery).

Observe that there are $pvetoes_{(C,V)}(c) - \ell$ edges according to the first edge group and exactly 3ℓ edges corresponding to the second voter group in the cover. Our constructed edge cover includes a total of $pvetoes_{(C,V)}(c) + 2\ell$ edges and we have $\iota(c_j) := \alpha(c_j) + \beta(c_j) \geq b_l(c_j)$ due to our assumption that c_j has at least as many vetoes as c in the final election, where $\iota(c_j)$ is defined as the number of edges incident to c_j in the cover. Since exactly 3ℓ edges are incident to b and since the capacity constraints for $*$ and $**$ are trivially satisfied, the first direction follows.

($\Leftarrow$): Now suppose that an edge cover defined as above exists. Again, let $\iota(x)$ be the number of edges incident to vertex x in this cover. Observe that $\iota(b) = 3\ell$ and $\iota(c_j) \geq b_l(c_j)$ ($1 \leq j \leq m-1$) according to the cover. The capacity restrictions of $*$ and $**$ are trivially satisfied. Moreover, the cardinality of the matching is $pvetoes_{(C,V)}(c) + 2\ell$, where 3ℓ edges are incident to b due to the capacity constraints of b. We define the following YES instance of 3-VETO-X-NECESSARY BRIBERY ($X \in \{\text{FP,PC}\}$):

- Each voter in $V^{\neg c}$ remains unchanged.

- According to the capacity of the cover, there are exactly $pvetoes_{(C,V)}(c) - \ell$ edges $\{c_i, c_j\}$, $\{c_i, *\}$, or $\{*, **\}$ in the cover (where $1 \leq i, j \leq m-1$, $i \neq j$). The briber leaves all these voters in V^c unchanged and bribes the other ℓ voters potentially vetoing c.

- Exactly $\beta(c_j)$ bribed voters veto c_j after the bribery ($1 \leq j \leq m-1$), where $\beta(c_j)$ denotes the number of edges $\{b, c_j\}$ in the cover. Since it holds $\beta(c_j) \leq \ell$ for each c_j and $\beta(c_1) + \ldots + \beta(c_{m-1}) = 3\ell$, Lemma 3.24 implies that the ℓ bribed voters can feasibly assign these 3ℓ vetoes to non-distinguished candidates without any c_j being vetoed at least twice by the same voter.

It follows that c is a necessary winner in the final election: We have $pvetoes_{(C,\bar{V})}(c) = pvetoes_{(C,V)}(c) - \ell$ voters potentially vetoing c after the bribery. Note that each c_j with $b_l(c_j) = 0$ does not beat c as enough voters in $V^{\neg c}$ definitely veto c_j and therefore c_j does not require any additional vetoes from voters in V^c, regardless of whether these voters are bribed or not. Now consider candidates with $b_l(c_j) > 0$. Since the edge cover is feasible, it holds $vetoes_{(C,\bar{V})}(c_j) =$

$vetoes_{(C,V^{\neg c})}(c_j) + \iota(c_j) \geq vetoes_{(C,V^{\neg c})}(c_j) + b_l(c_j) = vetoes_{(C,V^{\neg c})}(c_j) + \max(0, pvetoes_{(C,V)}(c) - \ell - vetoes_{(C,V^{\neg c})}(c_j)) = pvetoes_{(C,V)}(c) - \ell = pvetoes_{(C,\bar{V})}(c)$. Consequently, c beats or ties with c_j in the final election for every extension, and since this holds for each c_j, the distinguished candidate c is a necessary winner after the bribery of ℓ voters.

As computing a minimum generalized b-edge cover is in P and we can reduce our bribery problem to edge cover in polynomial time, our original problem is in P. ❑

Note that we used a construction and idea similar to the proof under full information. [17]

Last but not least, bribery in k-Veto is NP-complete for $k \geq 4$.

Corollary 3.30. *k-VETO-X-NECESSARY BRIBERY is NP-complete for every $k \geq 4$ and each $X \in$ PIM.*

This finding immediately follows Theorem 3.15 and from the fact that bribery under full information is hard [119].

3.6 Possible Bribery

In this section, our focus lies on the possible bribery problem which is a hybridization of standard bribery and possible winner (for the results, we refer to our work in [69]). The formal definition is as follows.

$\mathscr{F}$-X-POSSIBLE BRIBERY	
Given:	An election (C,V) with m candidates in candidate set C, n voters in voter set V according to $X \in \overline{\text{PIM}}$, a designated candidate $c \in C$, and a nonnegative integer ℓ.
Question:	Is it possible to change up to ℓ votes such that c is a winner of the election under $\mathscr{F}$ for at least one completion of the votes in V?

Possible bribery can be regarded as the optimistic variant of bribery under incomplete information. The question is whether there is a chance at all that the briber makes c a winner of the election via altering a given number of votes. If an instance of possible bribery leads to the output YES, this does not say anything about the quality of the designated candidate as this candidate might be a winner for almost all completions as well as for only one extension. Following this, possible bribery can be seen as some kind of minimum criterion for bribery under partial information.

Secondly, when we consider partial information as indecisiveness of voters, possible bribery can be regarded as a problem where the briber entirely convinces some (bribed) voters of his own ideas, and persuades the other voters (not bribed) to decide on one possible ranking (that is, completion of their partial vote), cf. also the arguing in [117, 146].

Possible bribery appears to be attractive to study from the theoretical point of view. Remind that possible winner under partial orders is only easy for Plurality and Veto, but hard for k-Approval

[17] We point out that Lin [119, 120] claimed that 3-VETO-BRIBERY is easy under full information for the co-winner model, but he did not provide any proof. As this proof had not been published, Lin's phd supervisor Edith Hemaspaandra sent a proof sketch to Gábor Erdélyi at the end of September 2015 while we were preparing our AAMAS paper [30]. However, their proof used a traditional b-edge cover approach, whereas we showed our result by means of *generalized* b-edge cover. Moreover, our proof is tailored to partial information, while Lin's proof is specific to full information. Nevertheless, our proof is strongly based on Lin's proof idea.

and k-Veto, for each $k > 1$ [21, 167]. Hence, our intuition initially was that similar, very general structures of partial information lead to hardness results, too, and we hoped for possible bribery to become hard for Plurality and/or Veto under partial orders. To put it another way, we study whether there are some problems for which both possible winner for the corresponding partial model and standard bribery are easy, but their combination is hard. For the necessary winner version of bribery, we have found many such problems, e.g., necessary bribery in Plurality for six models, among them Gaps and PC (cf. Section 3.5).

In the remainder of this section, we provide complexity results for the constructive variant of the possible bribery problem. The results are summarized in Table 3.6. We start with a theorem, pinpointing the connections between the possible bribery problem and the possible winner and standard bribery problems. (In each proof throughout this section, unless stated otherwise, our input is given as in the previous section about necessary bribery.)

Theorem 3.31. *$\mathscr{F}$-X-*POSSIBLE WINNER *and $\mathscr{F}$-*FI-BRIBERY *many-one reduce to $\mathscr{F}$-X-*POSSIBLE BRIBERY.

Proof. Let $X \in \overline{\text{PIM}}$. Observe that for $\ell = 0$, our problem is equivalent to $\mathscr{F}$-X-POSSIBLE-WINNER. For $X =$ FI (each vote is complete), possible bribery and standard bribery coincide. ❑

According to Theorem 3.31, we can immediately disclose some hardness results without any further considerations.

Voting rule	FI	Gaps	FP	TOS	PC	CEV	1TOS	1Gap	TTO	BTO
Plurality	P	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
2-Approval	P	**P**	**P**		*NPC*	**P**	**P**	**P**	**P**	**P**
k-Approval ($k \geq 3$)	NPC	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*
Veto	P	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
2-Veto	P	**P**	**P**		*NPC*	**P**	**P**	**P**	**P**	**P**
3-Veto	P	**P**	**P**	**NPC**	*NPC*	**P**		**P**	**P**	**P**
k-Veto ($k \geq 4$)	NPC	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*	*NPC*

Table 3.6: Summary of results for the possible bribery problem under the co-winner model. Key: P means "polynomial-time decidable", NPC stands for "NP-complete". Column FI displays the results for bribery under full information following from [79] for Plurality and Veto and from [119] for k-Approval and k-Veto ($k \geq 2$). Results in italic are hardness results that follow from already existing hardness results for bribery under full information or the possible winner problem. We exploit that the possible winner problem under partial orders is hard for 2-Approval [167], 2-Veto [21], and 3-Veto [21]. Moreover, 3-VETO-TOS-POSSIBLE WINNER is hard according to Theorem 3.13. Results in boldface are new.

Our first complexity result states that possible bribery in Plurality is easy for all nine models in PIM. Membership in P is shown via computing a maximum integral flow.

Theorem 3.32. PLURALITY-X-POSSIBLE BRIBERY *is in* P *for every model $X \in$ PIM.*

Proof. It suffices to regard PC and FP. The other structures of partial information are special cases of PC or FP, thus membership in P immediately follows for them. The proofs are identical for both models apart from the preprocessing. For FP, we assume that all voters approve of c who (1) rank

c first or (2) do not assign a fixed position to c and the first position is open. In all other votes, one or more non-distinguished candidates are potentially approved. Likewise, given partial orders, we assume that all those voters rank c first who do not definitely prefer any c_j over c. In all other votes v, the potential scorers are those candidates c_j for whom there exists no other candidate that v definitely prefers over c_j.

In a nutshell, for both models a voter either favors c (at least for some extension), definitely ranks some c_j first, or prefers one out of several distinct $c_{j_1},\ldots,c_{j_r}$, with $1 \leq j_1 < \ldots < j_r \leq m-1$ and $r \in \{2,\ldots,m-1\}$. We count all potential points for c, including both definite and uncertain approvals. In case there is an extension for which c is a winner, there is necessarily an extension with c being a winner gaining as many points as possible, according to the partial information given. This is a mere consequence of the weak monotonicity property holding for the Plurality rule.

First of all, if $pscore(c)+\ell \geq \frac{n}{2}$, we accept because in some extensions the briber can ensure that at least half of all voters give a point to c. Since no other candidate can achieve more points for such completions, c is a possible winner.

For $pscore(c)+\ell < \frac{n}{2}$, we aim at finding $n-pscore(c)-\ell$ voters definitely not ranking c first and complete them in a way that each c_j has at most $pscore(c)+\ell$ points—the best-case score of the c after the bribery. To do so, we define the following flow maximization problem. (For our purposes, we assume that w.l.o.g. exactly the voters $v_1,\ldots,v_{n-pscore(c)}$ definitely disapprove of c. All remaining voters potentially have c as their top preference and are thus assumed to vote for c.)

Our flow network is defined by its source x, sink y, its vertices $v_1,\ldots,v_{n-pscore(c)}$ and $c_1,\ldots,c_{m-1}$, and the following edges:

- There is an edge from x to v_i with capacity one ($i = 1,\ldots,n-pscore(c)$).
- There is an edge between v_i and c_j with capacity one if and only if c_j is potentially ranked first by v_i ($i = 1,\ldots,n-pscore(c), j = 1,\ldots,m-1$).
- There is an edge going from c_j to y with capacity $pscore(c)+\ell$ ($j = 1,\ldots,m-1$).

The first two edge groups guarantee that every voter not voting for c votes for one c_j. In contrast, the third group of edges ensure that—if possible—no candidate may reach a higher score than c for an extension with c being a winner.

We claim that the briber can make c a possible winner by bribing ℓ voters if and only if there is a maximum integral flow with value of at least $n-pscore(c)-\ell$.

($\Rightarrow$): Suppose that there is a successful bribery. This implies that we can find $n-pscore(c)-\ell$ voters who are (1) left unchanged by the briber and (2) certainly disapprove of c, complete these votes and every c_j has at most as many points as c after the bribery for this extension, that is, a score of $pscore(c)+\ell$. According to these aspects, we construct a flow F as follows. If voter v_i ($i \in [n-pscore(c)]$) remains unchanged and approves of c_j for the extension with c as a winner, set $F((x,v_i)) = F((v_i,c_j)) = 1$. Since there are $n-pscore(c)-\ell$ such voters, a flow with this size starts from the source x. Set $F((c_j,y)) = \sum_{i=1}^{n-pscore(c)} F((v_i,c_j))$, that is, $F((c_j,y))$ is equal to the number of voters approving of c_j in our given extension with c being a winner. As c is a winner, we have $F((c_j,y)) \leq pscore(c)+\ell = cap((c_j,y))$ for each $j \in [m-1]$.

Observe that all other capacity constraints are met, too. We have $F((x,v_i)), F((v_i,c_j)) \in \{0,1\}$ for each $i \in [n-pscore(c)]$ and $j \in [m-1]$. Hence, our flow F is feasible, integral, and has value $n-pscore(c)-\ell$.

($\Leftarrow$): Now suppose that an integral flow with a value of at least $n - pscore(c) - \ell$ exists. We define the following YES instance of possible bribery:

- Select $n - pscore(c) - \ell$ voters v_i according to edges with $F((x, v_i)) = 1$.
- The remaining ℓ voters are bribed and approve of c after the bribery.
- W.l.o.g., the first $n - pscore(c) - \ell$ voters v_i with $F((x, v_i)) = 1$ remain unchanged in our constructed extension (i.e., the voters with smallest index $i \in \{1, \ldots, n - pscore(c)\}$; this is only relevant when the maximum flow is larger than $n - pscore(c) - \ell$). For these voters v_i, it holds $F((v_i, c_j)) = 1$ for one non-distinguished candidate c_j, due to the flow conservation conditions. Hence, voter v_i approves of c_j in our given extension according to the flow.
- As the flow is feasible, we have $\sum_{i=1}^{n-pscore(c)} F((v_i, c_j)) = F((c_j, y)) \leq cap((c_j, y)) = pscore(c) + \ell$ for each $j \in [m-1]$, that is, every c_j has no more points than c in the final election.

Since we may consider approvals for non-distinguished candidates in $n - pscore(c) - \ell$ voters definitely not supporting c and all c_j have $pscore(c) + \ell$ or fewer points, c is a winner with $pscore(c) + \ell$ points for our constructed extension.

Because computing a maximum integral flow and transforming our initial problem to flow maximization are easy, our overall problem is in P. ❑

As we could observe, the bribery problem combined with possible winner is easy for all nine models in PIM. Verify that this proof generalizes the proof of PLURALITY-PC-POSSIBLE WINNER in [21] in a sense that for $\ell = 0$ both proofs more or less coincide. Figure 3.3 provides an example with $n = 5$, $pscore(c) = \ell = 1$, and $m = 4$. Voter v_1 definitely favors c_1, v_2 potentially ranks c_1 and c_2 on top, v_3 certainly likes c_2 most, v_4 potentially approves of c_2 and c_3, while v_5 potentially prefers c (we fix this approval, no matter whether this approval is definite or unsure). Each edge in the flow network is labeled with two numbers. The left number represents the number of flow units flowing through the network, the right number corresponds to the capacity of the edge. The flow of size $n - pscore(c) - \ell = 3$ and its corresponding edges are marked red. Notice that the briber bribes the voter v_3 and makes him approve of c. In our constructed extension associated with the flow F, voter v_1 favors c_1, whereas v_2 and v_4 vote for c_2.

For 2-Approval, our problem becomes hard under partial orders.

Corollary 3.33. 2-APPROVAL-PC-POSSIBLE BRIBERY *is* NP*-complete.*

Proof. Hardness follows from Theorem 3.31 and the fact that 2-APPROVAL-PC-POSSIBLE WINNER is NP-complete [167]. ❑

Opposed to our initial expectation, other very general partial information models do not yield hard possible bribery/winner problems. In fact, according to the next two theorems, possible bribery in 2-Approval is easy for seven out of nine structures. In contrast to bribery under full information (cf. [119]), the proofs for possible bribery do not work with b-edge cover, but with generalized b-edge matching (with upper and lower capacity restrictions both for edges and vertices) and require more elaborate constructions.

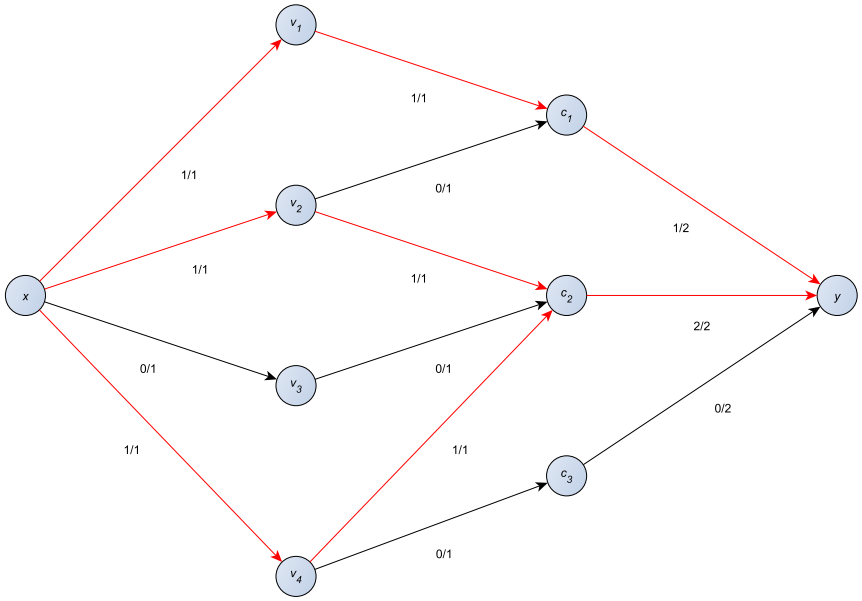

Figure 3.3: Example of the flow algorithm in the proof of PLURALITY-X-POSSIBLE BRIBERY.

Theorem 3.34. 2-APPROVAL-1TOS-POSSIBLE BRIBERY *is in* P.

Proof. Let $\tilde{C} \subseteq C$ denote the candidate set totally ordered by each voter. Our algorithm checks the following cases:

- $|\tilde{C}| \leq 2$ or $c \in C \setminus \tilde{C}$. Then there are extensions such that c is approved of by every voter. If $c \in C \setminus \tilde{C}$, there are completions where each voter ranks c first. In case $c \in \tilde{C}$ and $|\tilde{C}| \leq 2$, there are extensions for which all candidates in $C \setminus \tilde{C}$ are ranked behind the candidates in $\tilde{C}$ including c. We accept in both cases.

- $\tilde{C} = C$. In this case, we have full information for which the P result follows from [119].

- $c \in \tilde{C}$ and $C \setminus \tilde{C} = \{p\}$. In this case, we have $pscore(c) = score_{(\tilde{C},V)}(c)$. In other words, we account for all potential approvals for c. Each other candidate $c' \in \tilde{C}$ has at least $s_1(c')$ approvals and at most $s_1(c') + s_2(c')$, where $s_t(c')$ $(t = 1,2)$ denotes the number of voters ranking c' on position t when we restrict ourselves to candidates in $\tilde{C}$. Note that in each vote p is potentially, but never definitely approved of. We assume that voters regarding c as their second best choice in $\tilde{C}$ actually disapprove of p because otherwise c would be disapproved by these voters.

  First of all, if $pscore(c) + \ell \geq n$, we accept as the briber can ensure a full approval score for c in at least one extension. Thus, let $pscore(c) + \ell < n$. We transform our initial problem to a generalized b-edge matching problem defined below. For practical reasons, let $v_1, \ldots, v_{pscore(c)}$ be the voters ranking c among the first two positions in $(\tilde{C},V)$ and $v_{pscore(c)+1}, \ldots, v_n$ denote the voters definitely disapproving of c. Our matching problem is defined as follows. We are given an undirected multigraph $G = (\mathcal{V},\mathcal{E})$ with vertices

$\mathscr{V} = (C \setminus \{c\}) \dot{\cup} \{v_1, \ldots, v_{pscore(c)}\} \dot{\cup} K \dot{\cup} \{b\}$, where K is a set of auxiliary vertices. In other words, every non-distinguished candidate and all voters potentially approving of c yield a vertex in the graph each. We further create a vertex b representing the ℓ approvals that bribed voters assign to non-distinguished candidates after the bribery; remind that each bribed voter approves of c and another candidate. Last but not least, each voter v_i disapproving of c yields two vertices $k_1^{v_i}$ and $k_2^{v_i}$ in K ($i \in \{pscore(c)+1, \ldots, n\}$). The edges are defined as follows:

- For each voter v_i ($1 \leq i \leq pscore(c)$) favoring c over all other candidates in $\tilde{C}$ and considering c' as their second best choice within $\tilde{C}$, there are two edges $\{v_i, c'\}$ and $\{v_i, p\}$. If v_i ranks c' and c first and second restricted to $\tilde{C}$, respectively, there is an edge $\{v_i, c'\}$ (and no edge $\{v_i, p\}$ since p must not be approved by these voters in order that c's approval is accounted for).
- For each $d \in C \setminus \{c\}$, there are ℓ edges connecting d with b. [18]
- Each voter v_i ($pscore(c)+1 \leq i \leq n$)—that is, v_i surely disapproves of c—yields four edges $\{c', k_1^{v_i}\}$, $\{k_1^{v_i}, k_2^{v_i}\}$, $\{k_2^{v_i}, c''\}$, and $\{k_2^{v_i}, p\}$, where c' and c'' are voter v_i's most and second most preferred alternatives in $\tilde{C}$, respectively.

The capacities are as follows: $b_l(v_i) = b_u(v_i) = 1$ ($1 \leq i \leq pscore(c)$), $b_u(d) = pscore(c) + \ell$ ($d \in C \setminus \{c\}$), $b_l(b) = b_u(b) = \ell$, and $b_l(k_h^{v_i}) = b_u(k_h^{v_i}) = 1$ ($pscore(c)+1 \leq i \leq n$, $h = 1, 2$). All lower capacities not listed here are meant to be equal to zero.

Henceforth, we will refer to the latter group of edges, incident to some vertex in K, as *Construction I*.

The capacity restrictions for the vertexes v_i ($i \leq pscore(c)$) are to ensure that each of these voters approves of one other candidate in a given extension. The capacities $b_u(d)$ must be satisfied since otherwise some d beats c in our extension. Finally, b has the function to represent ℓ approvals that bribed voters assign to candidates other than c after the bribery.

We claim that the briber can make c a possible winner by bribing (w.l.o.g. exactly) ℓ voters if and only if the graph yields a matching with at least $2n - pscore(c)$ edges.

($\Rightarrow$): Suppose that the briber can make c a possible winner by bribing ℓ voters. Then c has a potential score of $pscore(c) + \ell$ and we can extend the votes not changed such that each other candidate d has no more points than c. In the following, we define a collection of edges according to our extension with c being a winner and argue at the end that this edge collection corresponds to a matching with the desired properties. First of all, we point out the following aspects. Each candidate $d \neq c$ receives (1) $\tau(d)$ approvals from the voters v_i ($1 \leq i \leq pscore(c)$) who approve of c as well (and who are not bribed by a rational briber), (2) $\gamma(d)$ approvals from the voters v_i ($i > pscore(c)$) left unchanged, and (3) $\beta(d)$ approvals from bribed voters after the bribery. Observe that the previous aspect suggests that V is partitioned into the following three voter groups: (1) voters approving of c before and after the bribery,

[18] Lin [119] defined similar edges in his proof for bribery in 3-Veto under full information (as already mentioned, this proof has never been published). Independently from Lin, we defined analogous edges (in form of capacitated edges) in our flow maximization algorithm in the proof of Theorem 3.40. This latter proof had been finished around one month earlier before we learned about Lin's proof in form of a private communication of Edith Hemaspaandra with Gábor Erdéyi at the end of September 2015.

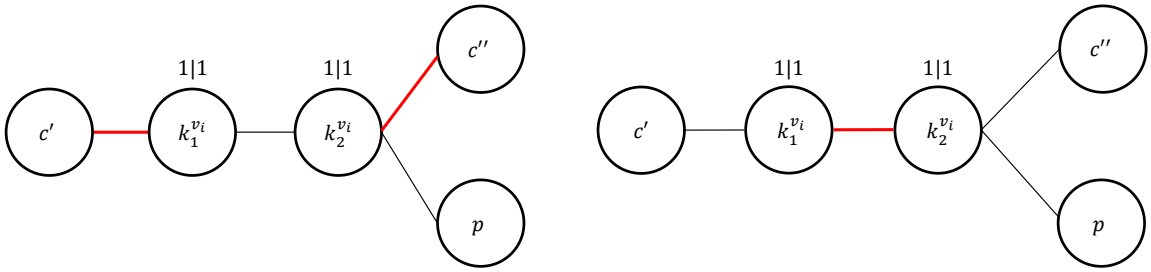

Figure 3.4: Example of a voter v_i ($pscore(c) < i \leq n$) according to Construction I. v_i ranks c' first and c'' second restricted to the candidates in $\tilde{C}$, where $c' \neq c \neq c''$ holds. The lower and upper capacities of a vertex in K, separated by a "|", are written above the vertex. The red edges in both graphs display two possibilities of which edges can belong to a matching according to vote v_i. Note that in the left graph the edge $\{k_2^{v_i}, p\}$ could be in the matching instead of the edge $\{k_2^{v_i}, c''\}$.

(2) voters disapproving of c before and after the bribery, and (3) voters disapproving of c before and approving of c after the bribery. Notice that there are $pscore(c)$ voters in the first group, $n - pscore(c) - \ell$ voters in the second group, and ℓ voters in the third group.

With these preconsiderations in mind, we define the following edge collection for our graph:

- Each voter v_i ($1 \leq i \leq pscore(c)$) in the first group yields one edge $\{v_i, c'\}$ or $\{v_i, p\}$ in our edge selection—depending on whether v_i approves of both c and c' or both c and p. Our matching construction ensures that p is never approved whenever $pos_{\tilde{C}}(c, v_i) = 2$, as this would exclude the approval for c. Note that there is a total of $pscore(c)$ edges according to the first group.
- Each voter v_i ($pscore(c) < i \leq n$) in the second group yields the two edges $\{c', k_1^{v_i}\}$ and either $\{k_2^{v_i}, p\}$ or $\{k_2^{v_i}, c''\}$—according as v_i approves of both c' and p or both c' and c''.
- Each voter v_i ($pscore(c) < i \leq n$) in the third group yields two edges $\{k_1^{v_i}, k_2^{v_i}\}$ and $\{b, d\}$, where v_i approves of $d \neq c$ and c after the bribery. (Notice that the approvals for non-distinguished candidates before the bribery do not matter and hence the edge $\{k_1^{v_i}, k_2^{v_i}\}$ is in the matching.

According to our bribery instance, each voter in the first group corresponds to one edge and all other voters correspond to two edges each in our collection. Hence, we obtain exactly $pscore(c) + 2(n - pscore(c) - \ell) + 2\ell = 2n - pscore(c)$ edges. Observe further that all capacity constraints hold. Each voter v_i ($1 \leq i \leq pscore(c)$) satisfies $\iota(v_i) := b_l(v_i) = b_u(v_i) = 1$ (where $\iota(x)$ is defined as the number of edges incident to vertex $x \in \mathcal{V}$ in our edge selection) as exactly one candidate is approved aside from c. Moreover, since ℓ voters are bribed and since bribed voters assign ℓ approvals to non-distinguished candidates, we have $\iota(b) = \sum_{d \in C \setminus \{c\}} \beta(d) = \ell$. Notice that $\iota(d) = \tau(d) + \gamma(d) + \beta(d) \leq b_u(d)$ for each candidate

d other than c (since d does not achieve a higher score than c). As the capacity constraints for the $k_h^{v_i}$ ($h = 1,2$, $pscore(c)+1 \leq i \leq n$) are met as well, our edge selection is a feasible matching and we are done.

($\Leftarrow$): Assume that our matching instance is a YES instance. We show that the briber can make c a winner for at least one completion by bribing ℓ voters. Given a matching, we construct the following possible bribery instance:

- Each voter v_i ($1 \leq i \leq pscore(c)$) either approves of c and one of the candidates p and $c' \in \tilde{C} \setminus \{c\}$, depending on whether the edge $\{v_i, p\}$ or $\{v_i, c'\}$ is in the matching.
- For w.l.o.g. the first [19] $n - pscore(c) - \ell$ voters for whom the edge $\{c', k_1^{v_i}\}$ and one of the edges $\{k_2^{v_i}, c''\}$ and $\{k_2^{v_i}, p\}$ ($pscore(c)+1 \leq i \leq n$) are in the matching, we assume that voter v_i approves of c' and either c'' or p, depending on whether the edge $\{k_2^{v_i}, c''\}$ or $\{k_2^{v_i}, p\}$ is in the matching.
- Whensoever the edge $\{k_1^{v_i}, k_2^{v_i}\}$ is in the matching, voter v_i is bribed.
- Note that there are $\ell_1 \leq \ell$ voters with edge $\{k_1^{v_i}, k_2^{v_i}\}$ in the matching (otherwise, the cardinality of this matching would be too small). We assume that the remaining $\ell - \ell_1$ voters with two edges ($\{c', k_1^{v_i}\}$ and either $\{k_2^{v_i}, c''\}$ or $\{k_2^{v_i}, p\}$) [20] in the matching are bribed as well. Since there are exactly ℓ edges incident to b in the matching, we let exactly $\beta(d)$ bribed voters approve of c and d after the bribery (where $d \in C \setminus \{c\}$ and $\beta(d)$ denotes the number of edges $\{b, d\}$ in the matching).

Observe that our matching yields $pscore(c)$ voters approving of c before the bribery, ℓ bribed voters (approving of c and one other candidate each), and $n - pscore(c) - \ell$ unchanged voters that disapprove of c before and after the bribery. As each candidate vertex d meets its upper capacity $pscore(c) + \ell$, it follows that c has the highest score among all candidates for our constructed completion of the resulting election and is hence a possible winner.

- $c \in \tilde{C}$, $2 \leq |C \setminus \tilde{C}| \leq m - 3$. In this case, there are at least two "outside" candidates and at least three candidates in $\tilde{C}$ (such that c does not necessarily have a full potential score in the best case). Generally, both first and second positions in subelection $(\tilde{C}, V)$ do not count in some votes (i.e., two candidates in $C \setminus \tilde{C}$ may be actually approved by a voter for a given extension), we nevertheless assume that all potential points count for c. Again we suppose that $pscore(c) + \ell < n$ as otherwise the briber can make c reach a full approval score for some extension and we trivially accept then.

  We point out that the "outside" candidates (let us denote them with p_j, $1 \leq j \leq r$, where $C \setminus \tilde{C} = \{p_1, \ldots, p_r\}$) may be regarded as a coalition of candidates who must not get more

[19] Note that the maximum matching may yield more than $n - pscore(c) - \ell$ voters v_i ($pscore(c) < i \leq n$) for whom two edges are in the matching. As we require only $n - pscore(c) - \ell$ voters disapproving of c before and after the bribery (and the remaining ones are bribed), we may restrict ourselves to the first $n - pscore(c) - \ell$ of these voters one-to-one corresponding to two edges incident to a vertex in K in the matching. These are w.l.o.g. the voters with the smallest index i. Observe that under this restriction all upper capacities $b_u(d)$ (with $d \in C \setminus \{c\}$) remain satisfied since we even consider fewer approvals for non-distinguished candidates then.

[20] that is, the voters v_i with largest index i and different from the $n - pscore(c) - \ell$ voters v_i ($i > pscore(c)$) left unchanged.

than $|C \setminus \tilde{C}| \cdot (pscore(c) + \ell)$ approvals in total. (In case the candidates in $C \setminus \tilde{C}$ receive exactly $|C \setminus \tilde{C}| \cdot (pscore(c) + \ell)$ approvals after the bribery for a given extension, it is possible that these approvals are distributed in a way that each $p \in C \setminus \tilde{C}$ has the same approval score as c.)

We start our analysis by guessing a pair (α_1, α_2) and check if c can be made a possible winner with ℓ voters being bribed for a given completion, at most α_2 of the unbribed voters approve of two p_j and at most α_1 voters in total approve of one p_j and one candidate in $\tilde{C}$ (possibly equal to c). Note that we may focus on all guesses with $\alpha_1, \alpha_2 \in \mathbb{N}_0$ and $2\alpha_2 + \alpha_1 = |C \setminus \tilde{C}| \cdot (pscore(c) + \ell)$ and $\alpha_2 \leq n - pscore(c) - \ell$. [21] All other cases, where the p_j may get more or fewer points in total, are not worth studying. Either the p_j might get too many points altogether and thus one among them necessarily beats c or the candidates in $\tilde{C} \setminus \{c\}$ get more points altogether (in case the p_j get fewer than $|C \setminus \tilde{C}| \cdot (pscore(c) + \ell)$ points in total).

To show membership in P, we borrow the matching algorithm from the previous case and adjust it as follows.

- Instead of p, there is a vertex $*$ with capacities $b_l(*) = 0$ and $b_u(*) = \alpha_1$. Vertex $*$ shall ensure that at most α_1 voters in the final election approve of exactly one p_j. These voters may approve of c as well, before and/or after the bribery. Note that voters approving of two p_j can only be found among those voters disapproving of c before and after the bribery.
- In place of edges incident to p, there are analogous edges incident to $*$.
- Again, voters ranking c first among the candidates in $\tilde{C}$ yield one of the edges $\{v_i, c'\}$ or $\{v_i, *\}$ ($c' \in \tilde{C} \setminus \{c\}$, $1 \leq i \leq pscore(c)$) in the matching.
- The voters according to Construction I correspond to the same edges as before on the whole, with one slight adjustment: voters v_i ($i > pscore(c)$) remaining in the election and approving of two p_j correspond to only one edge $\{k_1^{v_i}, k_2^{v_i}\}$. In contrast, ℓ bribed voters v_i correspond to an edge $\{k_1^{v_i}, k_2^{v_i}\}$ (with $i > pscore(c)$) and an edge $\{b, c'\}$ (with $c' \in \tilde{C} \setminus \{c\}$) or $\{b, *\}$ in the matching, all other voters again yield the edge $\{c', k_1^{v_i}\}$ and one of the edges $\{k_2^{v_i}, c''\}$ and $\{k_2^{v_i}, *\}$, for $c', c'' \in \tilde{C} \setminus \{c\}$.

Similarly to the case $C \setminus \tilde{C} = \{p\}$, the briber can make c a possible winner by bribing ℓ voters if and only if there is a feasible guess (α_1, α_2), defined as above for which a matching exists with $2n - pscore(c) - \alpha_2$ edges. Note that, to a certain extent, the α_2 votes—with two p_j being approved (instead of one or two candidates in $\tilde{C} \setminus \{c\}$)—can be considered as further "bribes" in order to decrease the scores of candidates in $\tilde{C} \setminus \{c\}$. The only difference is that they do not increase the score of c and two candidates from $C \setminus \tilde{C}$ are actually approved.

As computing a maximum matching is easy and there are only polynomially many guesses (α_1, α_2) to check in the latter subcase, our problem is in P. ❑

The approach with Construction I helped us handling the interdependencies characteristic of the 1TOS structure. The algorithm in the proof of Theorem 3.34 automatically guarantees that in

[21] The only voters that may approve of two candidates in $C \setminus \tilde{C}$ are voters not approving of c, before or after the bribery. There are at most $n - pscore(c) - \ell$ such voters.

case a voter's second best choice in $\tilde{C}$ is approved in a given extension, his favorite $\tilde{C}$ candidate is approved, too. Note that setting $\ell = 0$ yields an alternative proof of possible winner under 1TOS which has been proven by means of an entirely different approach in [33]. In their proof, however, c is meant to be a candidate in $\tilde{C}$. The 1TOS model, by contrast, allows c to be a new candidate, i.e., $c \notin \tilde{C}$. Yet their model is no real restriction—as the authors themselves point out in [33]—since the case $c \notin \tilde{C}$ directly implies that our instance is a YES instance.

Example 3.35 displays some numerical examples of the matching algorithm in the proof of Theorem 3.34.

Example 3.35.

(a) Let (C,V) be an election with candidate set $C = \{c,c',c'',p\}$, voter set $V = \{v_1,\ldots,v_4\}$, distinguished candidate c, and bribery limit $\ell = 1$. Each vote is partial according to the 1TOS model and $\tilde{C} = \{c,c',c''\}$ denotes the totally ordered subset. Finally, the voting rule is $\mathscr{F} =$ 2-Approval. The voters vote as follows:

$$v_1 : c' \succ c \succ c'', \quad v_2 : c'' \succ c' \succ c, \quad v_3, v_4 : c' \succ c'' \succ c.$$

(b) Let (C,V) be an election with candidate set $C = \{c,c',c'',c''',p_1,p_2\}$, voter set $V = \{v_1,v_2,v_3\}$, distinguished candidate c, and bribery limit $\ell = 1$. Each vote is partial according to the 1TOS model and $\tilde{C} = \{c,c',c'',c'''\}$ denotes the totally ordered subset. Moreover, the voting rule is $\mathscr{F} =$ 2-Approval. The voters' votes are as follows (we restrict the rankings to the voters' two favorite candidates in $\tilde{C}$):

$$v_1, v_2 : c' \succ c'', \quad v_3 : c' \succ c'''.$$

The detailed calculations can be found in the Appendix. Our next result says that the problem is easy for six other models.

Theorem 3.36. 2-APPROVAL-X-POSSIBLE BRIBERY *is in* P *for each model* $X \in \{Gaps, FP, CEV, 1Gap, TTO, BTO\}$

Proof. We may assume that c is fixed as high as possible and thus obtains all potential approvals. The following kinds of votes may occur (we let $1 \leq i,j \leq m-1$, $i \neq j$):

1. c and c_i are approved.
2. c is approved, no other candidate is definitely approved.
3. c_i and c_j are approved.
4. c_i is approved, the other approved candidate is not uniquely known, and c is certainly disapproved.
5. Both approval positions are open, and c is not approved.

A rational briber bribes only voters belonging to the latter three groups.

First, if $pscore(c)+\ell \geq n$, we immediately accept since the briber can reach a full approval score for c in some extensions. For $pscore(c)+\ell < n$, we argue as follows. In contrast to the previous case, c cannot reach a full score for any completion and, in particular, there are more than ℓ voters definitely not approving of c in the original election (C,V). We will transform our problem to an instance of generalized b-edge matching. In contrast to the proof for full information, we can no longer argue with the classical edge matching or cover approach as the fourth and fifth kinds of votes do not lead to any one-to-one correspondence between votes and edges, but produce bundles of edges or even complete subgraphs. The matching problem is defined below. For our purposes, we again let $v_1,\ldots,v_{pscore(c)}$ potentially approve c and $v_{pscore(c)+1},\ldots,v_n$ surely disapprove of c. We are given an undirected multigraph $G=(\mathscr{V},\mathscr{E})$ with vertices $\mathscr{V}=(C\setminus\{c\})\,\dot\cup\,\{v_1,\ldots,v_{pscore(c)}\}\,\dot\cup\,\{b\}\,\dot\cup\,K$, where K is a set of auxiliary vertices. Notice that all non-distinguished candidates and all voters potentially approving of c yield a vertex denoted by the same symbol each. The edges in $\mathscr{E}$ are as follows.

- Each voter v_i approving of c (i.e., $1\leq i\leq pscore(c)$) yields an edge between v_i and c_j ($1\leq j\leq m-1$) if and only if c_j is potentially approved of by v_i (aside from c).
- For each $j\in[m-1]$, there are ℓ edges $\{b,c_j\}$.
- Voters v_i ($pscore(c)+1\leq i\leq n$) certainly approving of c_j and possibly but not definitely approving of the pairwise different candidates $c_{j_1},\ldots,c_{j_r}$ ($2\leq r\leq m-2$) yield four vertices $k_h^{v_i}$ in K with capacities $b_l(k_h^{v_i})=b_u(k_h^{v_i})=1$ ($h=1,\ldots,4$). Moreover, there are the edges $\{c_j,k_1^{v_i}\}$, $\{k_h^{v_i},k_{h+1}^{v_i}\}$ ($1\leq h\leq 3$), and $\{k_4^{v_i},c_{j_t}\}$ ($1\leq t\leq r$).
- Voters v_i ($pscore(c)+1\leq i\leq n$) possibly but not definitely approving of the pairwise different candidates c_{j_t} ($1\leq t\leq r$, with $3\leq r\leq m-1$) yield the vertices $k_h^{v_i}$ ($1\leq h\leq 3$) and the edges $\{k_1^{v_i},k_2^{v_i}\}$, $\{k_1^{v_i},k_3^{v_i}\}$, $\{k_2^{v_i},k_3^{v_i}\}$, and $\{k_1^{v_i},c_{j_t}\}$ ($1\leq t\leq r$). The capacities are $b_l(k_h^{v_i})=b_u(k_h^{v_i})=1$ ($h=2,3$) and $b_l(k_1^{v_i})=b_u(k_1^{v_i})=2$.
- We obtain edges and vertices analogously to the previous case when voter v_i ($pscore(c)+1\leq i\leq n$) definitely approves of two non-distinguished candidates (the same setting as before with $r=2$).

We refer to the latter three kinds of edges as *Construction II*. The remaining capacities are given as follows: $b_u(v_i)=b_l(v_i)=1$, $1\leq i\leq pscore(c)$ (each of these voters approves of precisely one c_j), $b_u(c_j)=pscore(c)+\ell$, $1\leq j\leq m-1$ (each c_j must have no more points than c after the bribery for a completion with co-winner c), and $b_l(b)=b_u(b)=\ell$ (the bribed voters assign ℓ approvals to non-distinguished candidates after the bribery). All other lower capacity constraints are meant to be zero.

Figure 3.5 illustrates the edges according to Construction II and which edges can belong to a matching.

According to this construction, the voters v_i ($i>pscore(c)$) yield either two or three edges in the matching. (Note that a matching always exists. We select arbitrary $pscore(c)$ edges $\{v_i,c_j\}$ in $\mathscr{E}$ ($1\leq i\leq pscore(c)$, $1\leq j\leq m-1$), arbitrary ℓ edges $\{b,c_j\}$, and for each v_i, $i>pscore(c)$, either the

two edges $\{k_1^{v_i}, k_2^{v_i}\}$ and $\{k_3^{v_i}, k_4^{v_i}\}$ (when voter v_i definitely approves of exactly one non-distinguished candidate before the bribery) or the two edges $\{k_1^{v_i}, k_2^{v_i}\}$ and $\{k_1^{v_i}, k_3^{v_i}\}$ (when v_i certainly approves of either zero or two non-distinguished candidates). Verify that this collection of edges satisfies all capacity constraints.)

Each voter v_i ($i \leq pscore(c)$) corresponds to exactly one edge in the matching. A feasible matching further contains exactly ℓ edges according to points assigned by bribed voters to non-distinguished candidates after the bribery. Finally, each voter disapproving of c before the bribery and approving of c after the bribery (that is, the voter is bribed) yields two edges according to Construction II—either the two edges $\{k_1^{v_i}, k_2^{v_i}\}$ and $\{k_3^{v_i}, k_4^{v_i}\}$ or the two edges $\{k_1^{v_i}, k_2^{v_i}\}$ and $\{k_1^{v_i}, k_3^{v_i}\}$. In case a voter disapproves of c before and after the bribery, a matching contains three edges according to Construction II (cf. Figure 3.5).

As a combined result, each matching has a cardinality somewhere between $2n - pscore(c) + \ell$ and $3n - 2pscore(c) + \ell$ (as each voter disapproving of c yields either two or three edges according to Construction II). By means of a similar reasoning to the proof of Theorem 3.34, we accept if and only if there is a maximum matching with at least $3n - 2pscore(c)$ edges as then and only then we can find (1) exactly $pscore(c)$ edges according to voters approving of c before and after the bribery, (2) at least $3(n - pscore(c) - \ell)$ edges according to voters disapproving of c before and after the bribery, (3) at most 2ℓ edges according to Construction II representing voters disapproving of c before the bribery and approving of c after the bribery (each of these edges is incident to two different vertices in K), and (4) exactly ℓ edges incident to b (the ℓ approvals that non-distinguished candidates receive from bribed voters after the bribery).

As computing a maximum matching is in P and as the transformation to this matching problem is surely polynomially bounded, our overall problem is in P. ❑

To the best of our knowledge, neither Construction I nor Construction II are existing in literature. Moreover, we are currently not aware of any proofs in COMSOC showing membership in P reducing from *generalized* b-edge matching/cover. Construction II can be considered as a refinement of Construction I since it additionally models voters approving of two out of three or more possibly but not definitely approved candidates with the same priority (i.e., in contrast to the algorithm in the proof of Theorem 3.34, there are no interdependencies in a sense that, for a given extension, a candidate is approved only when another candidate is approved, too). Although such interdependencies exist for the 1TOS model, the proof of Theorem 3.34 using Construction I exploits that the candidates in $C \setminus \tilde{C}$ can be treated as a coalition of nameless candidates whose total number of approvals divided by the cardinality of $C \setminus \tilde{C}$ must not exceed the final potential score of c (loosely speaking, the "outside candidates" are identical for every vote).

This is not the case for the TOS model, in general. We point out that the complexity for this structure is still open. Matching approaches as in the proofs of Theorem 3.34 and 3.36 have failed up to now. Although we conjecture that even the possible winner problem (and thus the possible bribery problem) is hard for 2-Approval under the TOS model, we could not adjust the hardness proof for partial orders (which can be found in [167]) in order to show hardness for the TOS model.

For $k \geq 3$, we can derive an overall hardness result:

Corollary 3.37. k-APPROVAL-X-POSSIBLE BRIBERY *is* NP*-complete for each* $k \geq 3$ *and each* $X \in$ *PIM.*

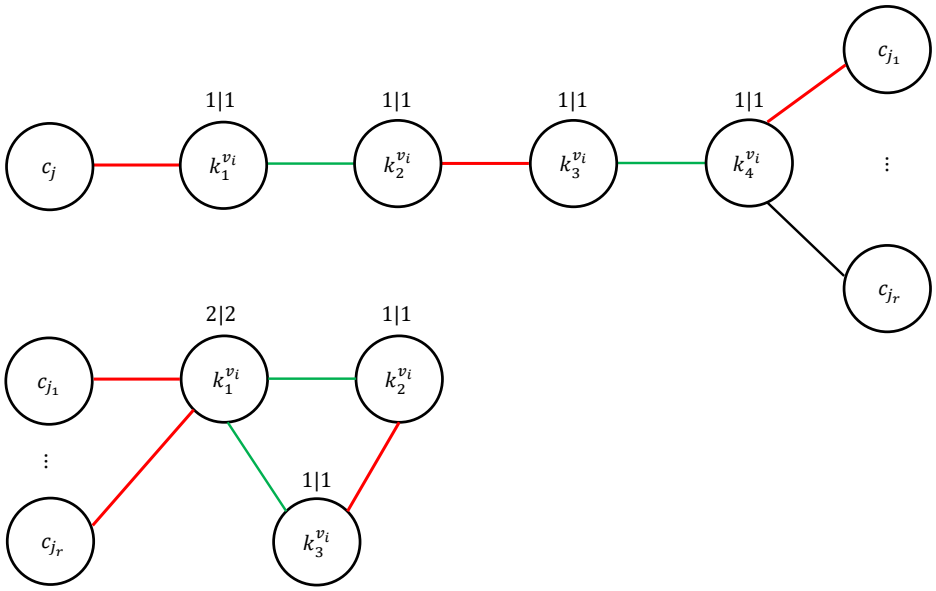

Figure 3.5: The upper graph describes the edges and vertices according to Construction II for voters definitely approving of exactly one non-distinguished candidate and possibly but not definitely approving one among two or more candidates other than c, provided the voter is not bribed. The lower graph corresponds to voters who definitely approve of either two or zero non-distinguished candidates, provided that the voter is left unchanged by the briber. Either the red edges or the green edges are in the matching. In the former variant, two approvals for non-distinguished candidates are accounted for, whereas in the second variant these approvals are not considered. Observe that for the first variant there is exactly one red edge $\{k_4^{v_i}, c_{j_t}\}$ $(1 \leq t \leq r)$ in the matching. Similarly, for the second variant, the red variant includes exactly two edges $\{k_1^{v}, c_{j_h}\}$ and $\{k_1^{v_i}, c_{j_s}\}$, where $1 \leq h < s \leq r$. Our example covers exactly one possibility for red edges incident to the c_{j_t}, $1 \leq t \leq r$ (namely, c_{j_1} and c_{j_r}). Again, the lower and upper capacities of a vertex in K are written above the vertex and separated by a "|".

Proof. Hardness immediately follows from Theorem 3.31 and because k-APPROVAL-BRIBERY (that is, under full information) is NP-complete for $k \geq 3$ [119]. ❑

Now let us turn to the k-Veto family. Pursuant to the next result, our problem is easy under all nine models in PIM:

Theorem 3.38. VETO-X-POSSIBLE BRIBERY *is in* P *for each model* $X \in PIM$.

Proof. We regard only PC and FP, the two most general models in PIM. Formally, for each vote v we can partition the candidate set C into a subset C_-^v containing only candidates definitely not vetoed and another subset C_+^v including all candidates potentially vetoed by v.

- For PC, we obtain $C_-^v = \{d \in C : \exists e \in C : d \succ_v e\}$ and $C_+^v = C \setminus C_-^v$.

- For FP, we first complete each vote with exactly position m being open by assigning this open position to the only candidate without any fixed position. After this preprocessing, there are two kinds of votes. Either position m is assigned to some candidate d or the veto position and at least one other position are open. In the former case, d is the unique veto candidate. In case of an open veto position, exactly the candidates not assigned to any fixed position are the potential veto candidates.

Note that these sets can be computed in $O(nm^2)$ time for both structures.

We premise that only definite vetoes count for c. Thus, in case $c \in C^v_+$ and $|C^v_+| \geq 2$, we reset $(C^v_-)' := C^v_- \cup \{c\}$ and $(C^v_+)' := C^v_+ \setminus \{c\}$ and consequently c gets only definite, unavoidable vetoes in some (best-case) extensions. Assuming the briber to be rational, he bribes only voters definitely vetoing c and he bribes as many such voters as possible. Let $\mathit{vetoes}(c)$ denote the number of certain vetoes for c.

First, let $\mathit{vetoes}(c) \leq \ell$. We accept since the briber can ensure zero vetoes for c for some extensions. Accordingly, bribing all voters vetoing c and making them veto arbitrary non-distinguished candidates is a successful bribery strategy.

If $\mathit{vetoes}(c) > \ell$, the briber can ensure that c has $\mathit{vetoes}(c) - \ell > 0$ vetoes for some extension. We show that our possible bribery instance is a YES instance if and only if c is a possible winner of the election $(C, \tilde{V})$ defined as follows.

- W.l.o.g. the voters $v_1, \ldots, v_\ell$ (the bribed voters in the original election) potentially veto all candidates in $C \setminus \{c\}$. [22]
- The voters $v_{\ell+1}, \ldots, v_{\mathit{vetoes}(c)}$ veto c.
- The remaining votes are partial according to PC (FP) and c is not vetoed.

The new election is equivalent to (C,V) in all votes not bribed, but contains ℓ voters potentially vetoing all non-distinguished candidates and definitely not vetoing c. These ℓ votes one-to-one correspond to ℓ bribed voters initially vetoing c for whom the briber can freely decide which voter among them vetoes which non-distinguished candidate. We point out that the transformation between these two elections is surely polynomial bounded.

Due to [167] and Theorem 3.8, possible winner is in P for Veto under the PC and the FP model, respectively. This in turn implicates that the possible bribery problem is in P for these models. ❑

Now let us focus on 2-Veto. In contrast to the Veto rule, we can settle a hardness result given partial orders:

Corollary 3.39. 2-VETO-PC-POSSIBLE BRIBERY *is* NP-*complete.*

Proof. We know from [21] that 2-VETO-PC-POSSIBLE WINNER is NP-complete. Hence, we may conclude from Theorem 3.31 that 2-VETO-PC-POSSIBLE BRIBERY is hard as well. ❑

Nevertheless, we obtain a P result for a total of seven models in PIM. Let us first show membership in P for six of these seven structures.

[22] All non-distinguished candidates are potentially vetoed as our algorithm must check first which candidate requires how many additional vetoes in order to make c a winner with $\mathit{vetoes}(c) - \ell$ vetoes.

Theorem 3.40. 2-VETO-X-POSSIBLE BRIBERY *is in* P *for every model* $X \in \{Gaps, 1Gap, FP, BTO, TTO, CEV\}$.

Proof. It suffices to regard $X \in \{\text{Gaps}, \text{FP}\}$. We rewrite each vote by fixing c on the highest possible position unless the position of c in a vote is already fixed.

The following types of votes may occur (with the veto candidates specified; it holds $1 \leq i, j \leq m-1$, $i \neq j$):

$$(c_i, c_j), \quad (c_i, *), \quad (*, *), \quad (c, c_i), \quad (c, *).$$

$*$ represents a veto which may go to different non-distinguished candidates for different extensions. In particular, a $*$ veto never goes to c in the rewritten election where all unsure vetoes for c are not accounted for. For voters of type $(*, *)$, no vetoed candidate is uniquely known.

In a nutshell, either two, one, or zero vetoes for the c_j are definite and the other zero, one, or two vetoes for the other non-distinguished candidates are uncertain. Or c is either vetoed together with some c_j or the other candidate vetoed aside from c is not uniquely known. Let $vetoes(c)$ be the number of definite vetoes for c.

First of all, if $vetoes(c) \leq \ell$, we accept since the briber can make c reach zero vetoes for some extension by bribing all voters vetoing c and making each of them veto two arbitrary other candidates.

Henceforth, let $vetoes(c) > \ell$. In this case, the briber bribes ℓ voters initially vetoing c and each of them does not veto c after the bribery (note that fewer bribes might suffice to make c a possible winner, but then ℓ bribes would lead to a successful bribery as well). Thus c has $vetoes(c) - \ell > 0$ vetoes after the bribery.

To show membership in P, we define the following flow network (for convenience, we assume that $v_1, \ldots, v_{n-vetoes(c)}$ do not definitely veto c and $w_1, \ldots, w_{vetoes(c)}$ definitely veto c):

The vertices include a source x, a sink y, vertices c_j $(j = 1, \ldots, m-1)$, $v_1, \ldots, v_{n-vetoes(c)}$, $w_1, \ldots, w_{vetoes(c)}$, k_i $(i = 1, \ldots, n - vetoes(c))$, and two additional vertices u and b. The edges are defined as follows:

- There is an edge from x to c_j $(j = 1, \ldots, m-1)$ with capacity $vetoes(c) - \ell$. These edges are to assure that every candidate receives at least as many vetoes as c, namely $vetoes(c) - \ell$.
- There is an edge (c_j, v_i) with capacity one if and only if c_j is definitely vetoed by v_i $(j = 1, \ldots, m-1, i = 1, \ldots, n - vetoes(c))$.
- There is an edge from c_j to k_i $(j = 1, \ldots, m-1, i = 1, \ldots, n - vetoes(c))$ with capacity one if and only if v_i possibly but not definitely vetoes c_j.
- There is an edge from k_i to v_i $(i = 1, \ldots, n - vetoes(c))$ with capacity equal to the number of indefinite veto positions in this vote v_i.
- There is an edge from c_j to w_i $(j = 1, \ldots, m-1, i = 1, \ldots, vetoes(c))$ with capacity one if and only if c_j is potentially vetoed by w_i. Note that c is the other candidate vetoed by w_i. [23]

[23] Notice that we do not need the auxiliary vertices k_i for the voters w_i as the algorithm either selects the candidate uniquely vetoed aside from c or one out of several candidates possibly but not definitely vetoed. In particular, the algorithm need not make a difference between definite and unsure vetoes for non-distinguished candidates in contrast to votes v_i where c is not vetoed.

- From v_i to y, there is an edge with capacity 2 ($i = 1,\dots,n - \mathit{vetoes}(c)$). These edges make sure that all voters vetoing two non-distinguished candidates may give only two vetoes to other candidates.

- From w_i to u, there is an edge with capacity 1 ($i = 1,\dots,\mathit{vetoes}(c)$).

- There is an edge from u to y with capacity $\mathit{vetoes}(c) - \ell$. Auxiliary vertex u shall guarantee that only $\mathit{vetoes}(c) - \ell$ of these votes (not bribed) initially vetoing c actually assign vetoes to non-distinguished candidates. The other ℓ voters originally vetoing c can arbitrarily give vetoes to non-distinguished candidates. These are modeled by b.

- From c_j to b, there is an edge with capacity ℓ ($j = 1,\dots,m-1$).

- From b to y, there is an edge with capacity 2ℓ. These previous two groups of edges ensure that each c_j receives at most ℓ vetoes from bribed voters, and bribed voters can give 2ℓ vetoes to non-distinguished candidates in total. The two capacity constraints are necessary and sufficient conditions for 2-Veto concerning how 2ℓ vetoes can be admissibly assigned to the c_j; in particular, these conditions ensure that no candidate is vetoed twice by the same voter (cf. Lemma 3.24).

We claim that there is an integral flow F with value $(m-1)\cdot(\mathit{vetoes}(c)-\ell)$ (that is, a maximum flow) if and only if the briber can make c a possible winner via bribing ℓ voters.

($\Rightarrow$): Suppose that a maximum flow F exists. We construct our completion with c being a winner after the bribery as follows. In a maximum flow, we have $F((x,c_j)) = \mathit{vetoes}(c) - \ell$ for each $j \in [m-1]$. Thus, each c_j has at least as many vetoes as c. These vetoes are distributed as follows. Each c_j, $j \in [m-1]$, receives $F((c_j,b)) \in \{0,\dots,\ell\}$ vetoes from the bribed voters after the bribery, according to the flow F. Which voters are bribed, will be explained below. Moreover, in case $F((c_j,k_i)) = 1$ or $F((c_j,v_i)) = 1$, voter v_i vetoes c_j ($1 \le i \le n - \mathit{vetoes}(c)$, $1 \le j \le m-1$). The same holds for voters w_i with $F((c_j,w_i)) = 1$ ($1 \le i \le \mathit{vetoes}(c)$) who remain unchanged. Since $\sum_{i=1}^{\mathit{vetoes}(c)} F((w_i,u)) \le \mathit{vetoes}(c) - \ell$, at most $\mathit{vetoes}(c) - \ell$ voters initially vetoing c and their vetoes for other candidates are actually considered.

Observe that the flow generally does not fill all veto positions for the c_j. We complete the remaining veto positions as follows:

- For all voters v_i with $F((v_i,y)) \le 1$ ($1 \le i \le n - \mathit{vetoes}(c)$), we complete the remaining $2 - F((v_i,y))$ veto positions in v_i's vote, left out by the flow, as follows. If $\mathit{cap}((c_j,v_i)) = 1$ and $F((c_j,v_i)) = 0$, let voter v_i veto c_j as v_i definitely vetoes c_j due to the definition of the flow network. Likewise, in case $F((k_i,v_i)) < \mathit{cap}((k_i,v_i))$ ($1 \le i \le n - \mathit{vetoes}(c)$), we suppose that v_i vetoes $\mathit{cap}((k_i,v_i)) - F((k_i,v_i))$ arbitrary candidates c_j with $F((c_j,k_i)) = 0$ and $\mathit{cap}((c_j,k_i)) = 1$.

- According to the maximum flow F, there are $\mathit{vetoes}(c) - \ell - \sum_{i=1}^{\mathit{vetoes}(c)} F((w_i,u)) (\ge 0)$ voters vetoing c who are not bribed and who are ignored by the flow (i.e., we have $F((w_i,u)) = 0$ and $\mathit{cap}((w_i,u)) = 1$). In other words, this number of voters w_i are left unchanged by the briber, in addition to the voters w_i with $F((w_i,u)) = 1$ since the maximum flow gets along with fewer

than $vetoes(c) - \ell$ voters vetoing c and their vetoes for non-distinguished candidates and yet each c_j has exactly $vetoes(c) - \ell$ vetoes in the final election according to the flow F.

We select arbitrary $vetoes(c) - \ell - \sum_{i=1}^{vetoes(c)} F((w_i,u))$ of these voters and let w_i veto any non-distinguished candidate c_j with $cap((c_j,w_i)) = 1$ and $F((c_j,w_i)) = 0$ (that is, c_j is potentially vetoed by w_i).

- The remaining voters w_i, not considered so far (that is, the voters w_i with $F((w_i,u)) = 0$ not among the $vetoes(c) - \ell - \sum_{i=1}^{vetoes(c)} F((w_i,u))$ voters w_i for whom we have assumed in the previous step that they remain unchanged, too), are bribed by the briber. Note that bribed voters assign 2ℓ vetoes to non-distinguished candidates in total and at most ℓ to each c_j. Hence, if $F((b,y)) < 2\ell$, we assign arbitrary $2\ell - F((b,y))$ vetoes to non-distinguished candidates such that each c_j has at most ℓ vetoes accordingly, taking into account that c_j already receives $F((c_j,b))$ vetoes from bribed voters, according to the flow. We refer to the algorithm in the proof of Lemma 3.24 describing which voter vetoes which candidates.

Notice that our constructed completion and the corresponding bribery leads to c being a winner in this completion.

($\Leftarrow$): Assume that the briber can make c a winner for some extension by bribing ℓ voters vetoing c. As c has $vetoes(c) - \ell$ vetoes after the bribery, each c_j has $vetoes(c) - \ell$ or more vetoes in this extension. For each c_j, we select $vetoes(c) - \ell$ arbitrary vetoes, either from bribed voters or from voters left unchanged. We construct a feasible flow as follows. First, set $F((x,c_j)) = vetoes(c) - \ell$ for each $j \in [m-1]$. Assign the flow units according to these voters as follows (we describe a recursive way to determine F):

1. If c_j ($1 \leq j \leq m-1$) is definitely vetoed by v_i ($1 \leq i \leq n - vetoes(c)$), set $F((c_j,v_i)) = 1$.
2. If c_j is possibly but not definitely vetoed by v_i ($1 \leq i \leq n - vetoes(c)$), set $F((c_j,k_i)) = 1$.
3. If c_j is potentially vetoed by w_i and w_i is left unchanged by the briber ($1 \leq i \leq vetoes(c)$), set $F((c_j,w_i)) = F((w_i,u)) = 1$.
4. If the $vetoes(c) - \ell$ vetoes meant for c_j come from exactly $\beta(c_j)$ bribed voters (after the bribery), set $F((c_j,b)) = \beta(c_j)$. Observe that $\beta(c_j) \leq \ell$ and $\sum_{j=1}^{m-1} \beta(c_j) \leq 2\ell$ holds.
5. Set further $F((b,y)) = \sum_{j=1}^{m-1} \beta(c_j) = \sum_{j=1}^{m-1} F((c_j,b))$.
6. Let $F((k_i,v_i)) = \sum_{j=1}^{m-1} F((c_j,k_i))$ ($1 \leq i \leq n - vetoes(c)$).
7. Compute $F((v_i,y)) = (\sum_{j=1}^{m-1} F((c_j,v_i))) + F((k_i,v_i))$ ($1 \leq i \leq n - vetoes(c)$).
8. Finally, set $F((u,y)) = \sum_{i=1}^{vetoes(c)} F((w_i,u))$.

Especially, the latter values exploit the flow conservation condition and "continue" the flow flowing through the source until all flow units reach the sink. Observe that the flow F is maximum and feasible as all constraints are satisfied.

As computing a maximum flow is in P and all constraints are bounded from above by a polynomial, the overall problem is in P. ❑

We achieve another easiness result for the 1TOS structure.

Theorem 3.41. 2-VETO-1TOS-POSSIBLE BRIBERY *is in* P.

Proof. Let $\tilde{C} \subseteq C$ be the subset completely ranked by each voter. Our algorithm distinguishes the following cases:

- $C = \tilde{C}$. Then our problem coincides with bribery under full information which is known to be easy according to [119].
- $c \in C \setminus \tilde{C}$ or $|\tilde{C}| \leq |C| - 2$. We accept as there are extensions of the original election where no voter vetoes c. Hence, c is a possible winner even without any voters being bribed.
- $c \in \tilde{C}$, $C \setminus \tilde{C} = \{p\}$. In this case, each candidate $c' \in \tilde{C}$ is definitely vetoed by the voters regarding c' as their least preferred alternative in $\tilde{C}$. Moreover, c' is possibly but not definitely vetoed by each voter preferring c' to exactly one candidate in $\tilde{C}$. In other words, last positions in subelection $(\tilde{C},V)$ definitely count as vetoes, whereas second to last positions in subelection $(\tilde{C},V)$ correspond to unsure vetoes in election (C,V). Note that each voter possibly but not definitely vetoes p. We assume that only definite vetoes count for c. Hence, in case c is a voter v's second to least preferred alternative in $\tilde{C}$, v's least preferred $\tilde{C}$ candidate and p are treated like definitely vetoed candidates.

  According to the previous reasoning, each voter either definitely vetoes two candidates or definitely vetoes one candidate and possibly but not definitely vetoes two candidates. Consequently, we may apply the same flow algorithm as in the proof of Theorem 3.40.

As all three cases are easy to decide, our overall problem is in P, too. ❑

In particular, Theorem 3.31 and Theorem 3.41 implicate that 2-VETO-1TOS-POSSIBLE WINNER is in P which has been claimed in Theorem 3.10 in Section 3.4.

As we could observe, possible bribery of the 2-Veto rule is easy for seven models and hard for one model. For the TOS model, we could not settle the complexity although we conjecture that even the possible winner problem is hard (similarly to 2-Approval).

For 3-Veto, possible bribery is hard for each structure in PIM under the unique-winner model:

Corollary 3.42. 3-VETO-X-POSSIBLE BRIBERY *is* NP*-complete for every* $X \in$ *PIM under the unique-winner model.*

Proof. Hardness follows from Theorem 3.27 and Theorem 3.31 as bribery under complete information is hard in the unique-winner model and possible bribery is a generalization of standard bribery. ❑

For the co-winner model, we cannot reduce bribery under full information in order to show hardness for possible bribery. Nevertheless, we obtain some hardness results by reducing from possible winner.

Corollary 3.43. 3-VETO-X-POSSIBLE BRIBERY *is* NP*-complete for every* $X \in \{TOS,PC\}$ *under the co-winner model.*

Proof. The problem 3-VETO-TOS-POSSIBLE BRIBERY is NP-complete due to Theorem 3.31 and 3.13. As 3-VETO-PC-POSSIBLE WINNER is NP-complete [21], 3-VETO-PC-POSSIBLE BRIBERY is hard as well, applying Theorem 3.31. (Note that we could also argue that PC is a generalization of TOS and therefore inherits the NP-hardness lower bound.) ❑

In contrast, our problem is easy for six models under the co-winner model. The proof showing polynomial-time decidability widely works with generalized b-edge cover.

Theorem 3.44. 3-VETO-X-POSSIBLE BRIBERY *is in* P *for each model* $X \in \{Gaps, FP, 1Gap, BTO, TTO, CEV, \}$ *in the co-winner model.*

Proof. It suffices to prove our theorem for $X \in \{\text{Gaps}, \text{FP}\}$. For both models, we merely count definite vetoes for c, unsure vetoes for c are not considered. Observe that for both structures, the following kinds of votes may exist (with the three vetoes specified, we have $1 \leq h, i, j \leq m-1$ and $|\{i,h,j\}| = 3$):

$$(c_h, c_i, c_j), \quad (c_h, c_i, *), \quad (c_h, *, *), \quad (*, *, *),$$

$$(c, c_h, c_i), \quad (c, c_h, *), \quad (c, *, *).$$

$*$ stands for an uncertain veto position for which the vetoed candidate may differ from completion to completion, and such a veto never goes to c (as only certain vetoes are supposed to count for c). Correspondingly, voters of the type $(c_h, *, *)$ definitely veto c_h and the other two vetoes are uncertain and not meant for c.

For $vetoes(c) \leq \ell$, we accept because the briber reaches that c is not vetoed for some extensions after the bribery.

In case $vetoes(c) > \ell$, the designated candidate c has a positive final veto number for each completion. We transform our problem to the generalized b-edge cover problem defined below. For our purposes, we let $V^{\neg c}$ denote the voters not definitely vetoing c (i.e., the veto for c is excluded due to our assumptions from above) and $V^c := V \setminus V^{\neg c}$ include all voters definitely vetoing c. For the sake of simplicity we assume that $V^{\neg c} = \{v_1, \ldots, v_{n-vetoes(c)}\}$ and $V^c = \{w_1, \ldots, w_{vetoes(c)}\}$. The cover problem is defined as follows. We are given an undirected multigraph $G = (\mathscr{V}, \mathscr{E})$ with vertices $\mathscr{V} = V^{\neg c} \dot\cup (C \setminus \{c\}) \dot\cup K \dot\cup \{b\}$, where K is a set of auxiliary vertices. The edges are defined as follows.

- For voter $v_i \in V^{\neg c}$, there is an edge $\{v_i, c_j\}$ if and only if v_i possibly but not definitely vetoes c_j.
- For $j \in [m-1]$, there are ℓ edges $\{c_j, b\}$ (c_j may receive at most ℓ vetoes from bribed voters).
- For each voter w_i ($1 \leq i \leq vetoes(c)$), definitely vetoing c and c_j, and potentially vetoing the candidates $c_{j_1}, \ldots, c_{j_r}$ (with $1 < r \leq m-2$), the graph contains four vertices $k_h^{w_i}$ in K ($1 \leq h \leq 4$) and the following edges according to Construction II: $\{c_j, k_1^{w_i}\}$, $\{k_1^{w_i}, k_2^{w_i}\}$, $\{k_2^{w_i}, k_3^{w_i}\}$, $\{k_3^{w_i}, k_4^{w_i}\}$, and $\{k_4^{w_i}, c_{j_t}\}$ ($t = 1, \ldots, r$). We have $b_l(k_h^{w_i}) = b_u(k_h^{w_i}) = 1$ for $h = 1, \ldots, 4$.

- Each voter w_i ($1 \leq i \leq \mathit{vetoes}(c)$) definitely vetoing c and possibly but not definitely vetoing the candidates $c_{j_1}, \ldots, c_{j_r}$ ($3 \leq r \leq m-1$; otherwise, all vetoed candidates would be unique) yields the three vertices $k_h^{w_i}$ in K ($1 \leq h \leq 3$) and the edges $\{k_1^{w_i}, c_{j_t}\}$ ($1 \leq t \leq r$), $\{k_1^{w_i}, k_2^{w_i}\}$, $\{k_1^{w_i}, k_3^{w_i}\}$, and $\{k_2^{w_i}, k_3^{w_i}\}$ according to Construction II. We have $b_l(k_h^{w_i}) = b_u(k_h^{w_i}) = 1$ ($h = 2,3$) and $b_l(k_1^{w_i}) = b_u(k_1^{w_i}) = 2$.

- Setting $r = 2$, we obtain the same edges and vertices as in the previous case representing voters definitely vetoing c, c_{j_1}, and c_{j_2}.

The vertex capacities not yet defined are $b_l(v_i) = b_u(v_i) = \alpha_i$ [24] for each $v_i \in V^{\neg c}$ (where $\alpha_i \in \{0,1,2,3\}$ denotes the number of indefinite veto positions in vote v_i; voter v_i assigns exactly α_i unsure vetoes to non-distinguished candidates), $b_l(c_j) = \max(0, \mathit{vetoes}(c) - \ell - \mathit{vetoes}_{(C,V^{\neg c})}(c_j))$, $1 \leq j \leq m-1$ (each c_j must have at least as many vetoes as c), and $b_l(b) = b_u(b) = 3\ell$ (bribed voters assign exactly 3ℓ vetoes to non-distinguished candidates). All remaining upper capacities are unlimited.

We point out the following aspects. Firstly, voters in $V^{\neg c}$ are never bribed and yield exactly $\sum_{i=1}^{n-\mathit{vetoes}(c)} \alpha_i =: \alpha$ edges in any feasible edge cover (if one exists). Secondly, a cover contains precisely 3ℓ edges incident to b, representing vetoes that non-distinguished candidates get from bribed voters after the bribery. Thirdly, each voter in V^c corresponds either to two edges (the voter is bribed [25]) or three edges (the voter is left unchanged and the vetoes for non-distinguished candidates count) according to Construction II in a cover.

Since we desire that at most $\mathit{vetoes}(c) - \ell$ of the voters in V^c may yield three edges in the cover (otherwise, we account for the vetoes for non-distinguished candidates in more than $\mathit{vetoes}(c) - \ell$ votes in V^c and hence effectively fewer than ℓ voters are bribed; yet the definition of our cover problem requires that exactly ℓ voters are bribed and c has precisely $\mathit{vetoes}(c) - \ell$ vetoes after the bribery), we search for a minimum cover with at most $(3\ell + \alpha) + (3(\mathit{vetoes}(c) - \ell) + 2\ell) = 3\mathit{vetoes}(c) + \alpha + 2\ell$ edges. ❑

The formal proof works very similarly to the one showing the result for 2-Approval (Theorem 3.36) which in turn uses a similar construction to the proof of Theorem 3.34. In contrast, we search for a minimum matching (i.e., a cover) in the case of 3-Veto, whereas we compute a maximum matching given 2-Approval. Both proofs use the same construction on the whole. While for 2-Approval we try to find as many voters as possible yielding three instead of two edges in the matching, for 3-Veto we seek to find as few voters as possible with three instead of two edges in the cover.

For 1TOS, the general complexity of possible bribery is still unknown to us in the co-winner model. However, we can settle a polynomial-time result for all cases but $c \in \tilde{C}$ and $|C \setminus \tilde{C}| = 2$. If $C = \tilde{C}$, the problem equals to bribery under full information. In case $c \notin \tilde{C}$ or $|\tilde{C}| \leq |C| - 3$, we accept as there are extensions for which c is not vetoed by any voter. Finally, in case $c \in \tilde{C}$ and

[24] Note that vertices v_i ($1 \leq i \leq n - \mathit{vetoes}(c)$) for which all three vetoed candidates are uniquely known, are isolated vertices with fixed capacity zero trivially satisfied by each cover. We introduced these vertices into our graph for the sake of compact representation, but we could also omit them.

[25] Actually, a bribed voter yields five edges in the cover—the two edges $\{k_1^{w_i}, k_2^{w_i}\}$ and $\{k_3^{w_i}, k_4^{w_i}\}$ (or $\{k_1^{w_i}, k_2^{w_i}\}$ and $\{k_1^{w_i}, k_3^{w_i}\}$) on the one hand and three edges incident to b and three different c_j each on the other hand.

$|C \setminus \tilde{C}| = 1$, easiness follows from transforming our problem to generalized b-edge cover, i.e., we borrow the algorithm from the proof of Theorem 3.44.

Finally, for $k \geq 4$ all problems are hard.

Corollary 3.45. *k-VETO-X-POSSIBLE BRIBERY is NP-complete for every $k \geq 4$ and every $X \in$ PIM.*

Proof. Possible bribery is hard for every model in PIM due to Theorem 3.31 and because bribery under full information is hard [119]. ❑

3.7 Conclusion

We have considered nine partial information models. Three models—FP, Gaps, and TOS—were introduced and formalized by us in [30], five models (PC, TTO, BTO, 1TOS, and 1Gap) had already been defined and studied in literature before, and the model CEV had been suggested, but not yet studied in the context of strategic behavior.

In a first step, we have pointed out the interrelationships between these models. Subsequently, we have analyzed the complexity of necessary and possible bribery. We have further closed all but two gaps for the possible/necessary winner problems. Our analysis is restricted to the voting rules k-Approval and k-Veto, to the constructive variants of the given problems, and basically to the co-winner model.

Concerning the possible/necessary winner problems, we could extend the result by [167] for necessary winner under partial orders via proving that necessary winner is easy for every scoring rule under the FP model as well. Hence, there is an overall polynomial-time result for all nine models in PIM and for every scoring rule. Moreover, we have pointed out that possible winner is easy for the whole families k-Approval and k-Veto under six of nine models. For k-VETO-1TOS-POSSIBLE WINNER, we have achieved a dichotomy result by proving the remaining cases $k \geq 2$. For the models PC, TOS, and 1TOS, nearly all possible winner problems are hard for the two voting rule families considered. This is no surprise, for these three models inhere a more intractable structure than the other six structures in PIM. Interestingly, ties matter for 3-VETO-1TOS-POSSIBLE WINNER which is easy under the co-winner model and hard under the unique-winner model.

With regard to strategic behavior, we have found that necessary bribery tends to result in harder problems under incomplete information compared to full information. As a byproduct, we have discovered several instances where ties matter. In other words, several problems are easy under the co-winner model and hard under the unique-winner model. Notice that in each of these cases the underlying voting rule is 3-Veto. In opposition to necessary bribery, we have not found any instance of $\mathcal{F}$-X-POSSIBLE BRIBERY for which $\mathcal{F}$-BRIBERY and $\mathcal{F}$-X-POSSIBLE WINNER are in P, but their hybridization is NP-complete. Therefore, additional hardness for possible bribery compared to bribery under full information follows only for problems for which the possible winner counterpart is already hard.

We point out that only k-Approval offers some additional worst-case barrier under incomplete information for necessary bribery, while for k-Veto the complexities remain unchanged compared to full information. It is worth mentioning that more or less the same models yield additional hardness

for k-Approval. Interestingly, the three models TTO, CEV, and 1TOS never increase the complexity for necessary bribery, compared to bribery under full information. By contrast, Gaps, 1Gap, FP, PC, TOS, and BTO reveal the same complexity-theoretic behavior in necessary bribery and hence all of them either yield a hard problem or a given problem is easy for all six of them.

Generally speaking, possible bribery does not satisfy our initial expectations that all problems become hard for the PC structure and for other general structures—such as Gaps or FP—as well. In contrast, necessary bribery meets our expectations concerning additional hardness under partial information, albeit merely for k-Approval and not for k-Veto. Hence, k-Approval offers some additional worst-case protection against bribery, not only for PC, but as well for other general structures such as Gaps, 1Gap, or FP. As pointed out, possible bribery is harder under partial information than bribery under full information, but merely because some possible winner problems are already hard.

One model worth further investigations is NI ("No Information") which was proposed in [39]. Under this structure, an external agent or a manipulator (group) has no information at all. This structure can be considered as the counterpart of FI ("Full Information") and is—just like FI—a special case of all models in PIM, but independent from FI. One can easily verify that necessary and possible bribery are easy for the whole families k-Approval/-Veto. For possible bribery, the P result is trivial even for every scoring rule. Even with a bribery limit $\ell = 0$, we may assume that all voters rank c first. In the co-winner model, this immediately implies that c is a possible winner. Under the unique-winner model, some additional cases have to be distinguished, but the problem is yet easy for all scoring rules. For necessary bribery, when restricting ourselves to k-Approval and k-Veto, one simply has to check whether c is a necessary winner when the briber assigns the approvals as uniformly as possible to non-distinguished candidates. We assume that all unbribed voters approve of (each) c_j and not c (similar arguing holds for k-Veto). It would be interesting to know whether necessary bribery is easy for all other scoring rules and—if not—for which classes of scoring rules.

By regarding nine different structures, we could observe interesting complexity differences between the models which exceed our initial expectations. So the hardness in possible bribery for 2-Approval and 2-Veto only holds for PC, but we could not settle hardness results for any other model. Although 1TOS is one of the most special models in PIM, almost all possible winner problems of the k-Approval/-Veto rules are hard for 1TOS, but easy for six other structures in PIM. In contrast, BORDA-1TOS-POSSIBLE WINNER is in P [32], but BORDA-X-POSSIBLE WINNER is hard for all other models in PIM. This holds since BORDA-CEV-POSSIBLE WINNER is identical to BORDA-X-COALITIONAL MANIPULATION which was shown to be hard in [23, 41]. As CEV is a special case of all models in PIM$\backslash\{$1TOS$\}$, hardness follows for the other models. The fact that BORDA-CEV-POSSIBLE WINNER is hard also yields an instance where CEV (a structure very simple and easy to handle) yields a hard problem, but its counterpart under full information, namely BORDA-WINNER, is easy. One can easily find other examples of model pairs where the first model yields a hard problem and the second structure yields an easy problem. We recently studied CCAV and CCDV under these nine models, both in a necessary and possible winner version [143, 70]. Necessary/possible control by adding voters, in the constructive version, asks whether a chair can add up to ℓ unregistered voters such that a distinguished candidate is a necessary/possible winner of the resulting election. We could show that TOS is the only model for which necessary constructive control by adding voters in k-Approval is easy for $k \leq 2$ and hard for $k \geq 3$ when the registered voters are complete and the unregistered voters are partial. For all other structures in PIM, the same problem is either easy exactly for $k = 1$ (namely, for PC, FP, 1Gap, and Gaps) or for $k \leq 3$ (for BTO,

1TOS, TTO, and CEV). For $k = 2$, the problem is easy for TOS and hard for Gaps, 1Gap, and FP. Hence, we could find a problem that is easier for TOS than for Gaps, 1Gap, and FP. Observe further that necessary bribery in k-Approval and k-Veto reveals the same complexity theoretic behavior for the BTO structure as for the models Gaps, 1Gap, FP, PC, and TOS, but this is not the case for our given control example. Such examples might encourage researchers to study these structures more closely and possibly find further models with real-world applications.

We have recently found two such models—FP1CS and 1SFP. Both models are defined as special cases of FP. While for FP1CS the candidate set with known fixed positions is the same for all voters (formally, $C^v =: \bar{C}$ for each $v \in V$), for 1SFP the set of known fixed positions is constant for all voters (formally, $Pos^v =: \overline{Pos}$ for every v). E.g., one can easily show that PLURALITY-FP1CS-NECESSARY BRIBERY is easy, while the same problem is hard for FP (all candidates in $C \setminus \bar{C}$ can be assumed to potentially score in all votes where no candidate in $\bar{C}$ is ranked first; hence, they can be treated like one candidate (coalition) and our problem reduces to bribery under full information to a certain extent). Considering these two structures in our detailed complexity analysis appears to be an intriguing task for future research.

We point out that the concept of worst-case complexity has some limitations. As the name suggests, hardness is only guaranteed in the worst-case. Hence, even when a problem is formally hard, this does not mean that the problem requires exponential running time in the average case (cf. the simplex algorithm [40] that is in P for typical-case instances; one has to construct some instances far from real-world applications in order to show hardness in the worst-case [116]). To read more about average case complexity in the context of voting, we refer to [66, 140]. Moreover, problems—though being NP-hard—can yet be approximated in polynomial time (by way of example, see [77] for some approximation algorithms in voting), easy for some special structures, or there might exist good heuristics to deal with a given problem (cf. the arguing in [89] for this and the next paragraphs about worst-case complexity).

Besides, when a problem is in $O(e^{0.0000001n})$ (where e is the Euler function), the instance size must be exorbitantly large that current computers meet their limits. By contrast, polynomial-time algorithms with running time $O(n^{100000})$ are formally easy, but do not work efficiently in practice.

Furthermore, referring to voting, hardness may hold for general preference profiles, but preference profiles are often structured according to a special pattern. For instance, in political elections profiles are often single-peaked or nearly single-peaked [74, 67]. Single-peaked preferences can be motivated as follows. Assume that there is an ideological spectrum, such as the left-right spectrum in political elections. Each voter favors one party placed on an axis and the further away another party is located on the left-right axis, the less the voter prefers the party. Nearly single-peaked profiles are "within a certain distance threshold" to a profile single-peaked with respect to a certain axis. Other structures, such as single-dipped [115, 2] or single-crossing preference profiles [29], exist as well. It would be nice to know whether our hardness results still hold when the voter profile exhibits some special structural property.

Last but not least, hard problems might become easy when some input parameter of a given problem is small. Concerning elections, small numbers of voters, candidates, and/or small bribing limits may simplify things to a malicious agent. This perspective encouraged several researchers to study the parameterized complexity for problems related to voting (we refer to Section 3.1 for some literature). Referring to partial information, a hard problem could become easy when the amount of existing/missing information is small. For instance, under the FP structure one could restrict the

number of known or unknown positions for every voter or in total.

Despite all its limitations, determining the worst-case complexity of a given problem appears to be a reasonable first step towards studying the qualitative properties of voting rules with regard to manipulative attacks. Accordingly, all these drawbacks of the concept of worst-case complexity pose many questions for future research.

We point out that even easy problems may offer a certain protection against undesired attacks when there is partial information. As an example consider instances for which (nearly) no information is given [39]. Even though it is easy to decide whether a necessary bribery exists or, more precisely, it can be verified in polynomial time that a necessary bribery does not exist, this information might be low-value to the briber. Consider an instance of bribery in Plurality for which the briber has no information at all. Observe that the briber must bribe at least half of all voters to ensure that his favorite candidate c is a necessary winner after the bribery. If his bribery limit is smaller, there may be extensions where one and the same candidate $d \neq c$ is ranked first by all voters not bribed. Note that $\ell \geq |V|/2$ always yields a YES instance for bribery in Plurality in general, regardless of the amount of given information. In contrast, if $\ell < |V|/2$, the briber cannot make c a necessary winner and therefore the lack of information provides a certain protection against bribing attacks. Admittedly, one may argue that necessary winner merely concerns the worst-case of a distinguished candidate. Accordingly, bribing fewer than half of all voters can lead to a sufficiently high probability that c is a winner after the bribery. In real-world settings, it is thinkable that all other voters spread their votes or even vote for c as well. Nevertheless, the lack of information may make things considerably harder to the briber.

For future research, we further refer to the open problems in this chapter. So many results have only been shown under the co-winner model and accordingly providing all proofs for the unique-winner model as well appears to be an intriguing task. In the following chapter, we will see that some proofs require additional considerations under the unique-winner model. One could also extend our study to other voting rules, other models (e.g., 1SFP and FP1CS), control, other bribery variants and restrictions, or manipulation (regard the models different from PC), to name a few. Besides, it would be nice to achieve dichotomy results for all scoring rules in all regarded problems. Other natural extensions of our research could be introducing weights and price tags for each voter.

Where we studied the constructive settings in this thesis, their destructive counterparts raise many questions as well. So we showed that *Possible Destructive Control* and *Possible Destructive Bribery* are easy even for all scoring rules as one has to check for at most $m-1$ non-distinguished candidates d whether an extension exists for which the despised candidate performs as badly as possible and d as well as possible (for detailed proofs, cf. [69, 70]; we omitted these proofs as in this thesis we study merely constructive problems in voting).

Another intriguing problem is *Dominating Bribery* defined in the style of the problem *Dominating Manipulation* in [39]. This problem asks whether the briber can bribe in a way that for each extension of the final election the briber is at least as satisfied with the outcome of the election as before the bribery, and there is at least one extension for which the briber is more satisfied than before. This definition was inspired by the well-known *Pareto Dominance* (for a formal definition, we refer to [146]).

In Section 3.1, we have mentioned the voting rules Normalized Range Voting and Fallback Voting which are currently the two voting rules with the smallest number of vulnerabilities among 22 selected control types (under full information). Since both voting rules are not exclusively

preference-based, it would be nice to know (1) whether one can find some partial information model tailored to these rules and (2) whether all control problems are either computationally intractable or impossible under partial information. Moreover, there might exist other voting rules displaying better or similar complexity theoretic behavior under complete or partial information.

For future research, we further refer to more refined concepts than possible/necessary winner to deal with partial information such as qualitative or probabilistic approaches. In this context, we point out that there are some works about applying voting rules directly to partial votes, such as votes with ties or top-truncated votes (cf. Section 3.1). A fascinating field of research could be combining necessary/possible winner with voting rules tailored to partial votes.

Other possible research directions include voting rules based on pairwise comparisons applied to voters with intransitive preferences which invite us to study several problems under the PC structure.

One could also study game-theoretic approaches (with two or more bribers or campaign managers) and/or combine bribery with manipulation and/or control (see [87] wherein a model combining manipulation and control was studied).

Chapter 4

Lot-based Voting

In this chapter, we regard a different way of voting under incomplete information. In contrast to Chapter 3, each voter declares a complete ranking over all candidates in C, but there is a lottery picking $K \in \{1,\dots,n\}$ voters at random and the voting rule is applied to these K voters. Although this general idea dates back to ancient Greece and was also used in a fairly complex process for the election of the Doge of Venice, similar ideas are used today in the selection process of the chair of the Internet Engineering Task Force (IETF). Furthermore, lot-based voting rules can describe elections where (at first) only K sample voters are actually accounted for. So, in political elections in some countries only a small sample of voters are considered and this sample is meant to be representative for the entire electorate.

One of the main motivations for introducing this additional non-deterministic step of selecting some voters at random is the hope that this step can impede strategic behavior and corruption.

Not so long ago Walsh and Xia [160] have defined lot-based voting and studied the computational complexity of evaluation as well as manipulation for voting rules such as Borda, STV [1], Copeland, and Maximin. According to their definition for lot-based voting in [160], the winner of an election is determined in two steps. First, K voters are selected by a lottery over the set of voters; second, a voting rule is applied on this K-voter subelection. Although several interesting questions in the field of lot-based voting were answered in their work, at the same time even more new questions were raised. One of these questions is the computation of possible and necessary winners (cf. [117]) in the setting of lot-based voting rules.

In this chapter we address this open question by analyzing the computational complexity of the evaluation (i.e., checking whether the winning probability of a given candidate exceeds a given bound), the possible and the necessary winner problems in the lot-based setting for k-Approval and k-Veto. Another natural question is to ask in how many subelections drawn by the random step a candidate is the unique winner. Analyzing the counting complexity of this problem is particularly relevant as solving this task is crucial for computing exact winning probabilities for a given candidate. Furthermore, we analyze the complexity of possible/necessary bribery, i.e., the setting where an external agent can influence one or multiple voters in order to make a designated candidate win for at least one/all size K subsets of the voters.

[1] The STV rule can be regarded as a generalization of Plurality with Runoff with at most $|C|-1$ rounds where in each step, the candidate with the smallest plurality score is ruled out. As soon as a candidate has an absolute majority of votes, this candidate is the winner of the election.

Our main contributions concerning lot-based voting are as follows:

- In our complexity analysis, we study the problems evaluation as well as possible and necessary winner for k-Approval and k-Veto in the lot-based setting. In particular, we obtain a complete picture for possible and necessary winner.
- We explore how hard it is to count the number of winning subelections and present a complete picture. For 2-Approval or 2-Veto, the decision problem is in P, but the counting problem is # P-complete, i.e., the task is easy to decide but hard to count.
- We also study the complexity of bribery. Unfortunately, for k-Approval and k-Veto the complexity of the lot-based setting matches the standard case. From the perspective of worst-case complexity, this indicates that further voting rules and lot-based formalisms have to be found to impede bribery.
- For 3-Veto we find several cases where ties matter, i.e., the complexity differs depending on whether the winner is required to be unique or can be tied with other candidates.
- On our way to prove our results for #LOTTHENk-VETO, we settle the complexity of VETO-#CCAV, namely membership in FP. (The counting complexity of control for Veto was left open by [164].)

4.1 Related Work

Our work was motivated by the paper by Walsh and Xia [160]. They explored the normative properties of lot-based rules. Furthermore, they introduced the evaluation problem for lot-based rules and investigated it as well as two manipulation problems from the complexity theoretic perspective for some prominent voting rules. In contrast, we introduce the necessary and possible winner problems for lot-based rules and study bribery as well. As pointed out in Section 3.1, the necessary and possible winner problems go back to Konczak and Lang [117]. There is, however, an important difference between their and our definitions. While Konczak and Lang defined their problems for incomplete profiles in a sense that even individual votes can be incomplete, we only allow incomplete profiles in a sense that we choose a fixed-size subelection with complete votes from the original election.

Lot-based voting rules can be seen as hybrid rules where we first use a lottery selecting the voters for a final round, to which we apply a certain voting rule. Conitzer and Sandholm [37] and Elkind and Lipmaa [57] studied somewhat similar hybrid systems from a complexity theoretic point of view. In their setting, in a first step a voting rule is used to rule out some candidates. In contrast, in lot-based voting rules in the first step some voters are ruled out randomly. Furthermore, the above mentioned two works did not investigate the possible and necessary winner problems for their voting rules. Hybrid voting rules have been further used in [106] and in [68, 63] for showing hardness in electoral control, however, these are very different settings compared to ours.

Lot-based voting rules are randomized voting rules introducing uncertainty to elections. In our case, the set of voters whose votes will count is not known beforehand. There are several papers handling uncertainty in elections. Elkind and Erdélyi [52] considered voting-rule uncertainty where the uncertainty is on the voting rule itself (it is picked from a set of possible voting rules after

all votes are cast). A similar model was introduced by [19]. There is much more literature about uncertainty in voting, cf. Section 3.1.

Our possible winner problem is somewhat related to the constructive control by adding voters (CCAV) problem introduced by [16]. Recall that in CCAV we are given an election, an additional set of voters, an integer ℓ, and a distinguished candidate. The question is whether we can add up to ℓ voters from a pool of additional voters to the election such that the distinguished candidate is the winner of the resulting election. One can consider our possible winner problem as a special case of CCAV where the original election consists of no voters and we want to add exactly a certain number of voters from the additional voters such that the distinguished candidate is the winner. This, however, has absolutely no impact on our complexity results as in CCAV we add up to a certain number of voters and in the possible winner problem we have to add *exactly* a given number of voters to an empty voter set. In fact, we will see that this small difference can lead to different complexity results. [2] To a certain extent, our problem is also related to the constructive control by deleting voters (CCDV) problem introduced by [16]. In CCDV, a chair deletes up to ℓ voters from the election. Our possible winner problem can be seen as a special case of CCDV where a chair deletes *exactly* $|V| - K$ voters from the election and at least one voter remains in the election. Similarly to CCAV, this (at first glance) little difference can yet lead to different complexity results between CCDV and the possible winner problem. In particular, whenever CCDV is in P for a given voting rule, this does not mean that the possible winner problem for the corresponding lot-based voting rule is in P as well. [3] We refer to [119] for an overview of the results of CCAV and CCDV in the k-Approval/-Veto families under complete information. [4] In contrast to Lin, we lay our focus on the unique-winner model. In the context of control, we have to mention the work by Wojtas and Faliszewski [164] who studied the counting complexity of CCAV in k-Approval, among others.

Our possible winner problem is a special case of a method for designing committees [58]. We inherit the results for k-Approval and for Veto (see Table 4.1).

We further point out the overview article in Chapter 1 in [60] and the references therein which are in particular about axiomatic properties in randomized social choice. Some works examine the mechanism *random dictatorship* [95] and its axiomatic properties. Loosely speaking, a voter is picked at random and the voter's most preferred candidate wins the election. These settings are different from ours.

Finally, as mentioned above, we have found some problems easy to decide but hard to count. More of such problems can be found in [157] and [164]. While the former article provides a detailed overview of various problems with easy decision and hard counting variant (especially graph-theoretical or combinatorial problems), the latter paper presents some easy to decide and hard to count problems related to voting.

---

[2] E.g., LOTTHEN3-APPROVAL-POSSIBLE WINNER [58] and 3-APPROVAL-CCAV [119] are hard and easy, respectively, while LOTTHEN3-VETO-POSSIBLE WINNER and 3-VETO-CCAV are easy and hard, respectively (cf. Theorem 4.5 and [119]). All results hold under the co-winner model.

[3] By way of example, LOTTHEN3-VETO-POSSIBLE WINNER is NP-complete due to Theorem 4.4 while 3-VETO-CCDV is in P [119]. Both results hold under the unique-winner model.

[4] For Plurality, the results originally follow from [16].

4.2 Problem Settings

Lot-based voting rules are defined as follows. For a given election (C,V), let K be a fixed (not constant, in general) natural number with $K \leq |V|$. For a voting rule $\mathscr{F}$, the randomized voting rule LotThen$\mathscr{F}$ selects the winner in two steps: first, K voters are selected uniformly at random, second, the winner is chosen from this K-voter subelection via the voting rule $\mathscr{F}$.

Throughout this chapter, our input is an election (C,V) with m candidates in candidate set $C = \{c_1,\ldots,c_{m-1},c\}$, n voters in voter set $V = \{v_1,\ldots,v_n\}$ (where $m,n \in \mathbb{N}$ [5]), a distinguished candidate $c \in C$, and a lot size $K \in \mathbb{N}$ with $K \leq n$. When regarding evaluation, $p \in [0,1](\cap\mathbb{Q})$ denotes a probability threshold, whereas for bribery problems a nonnegative integer ℓ with $\ell \leq n$ denotes the bribery limit. We often denote the final election with $(C,\bar{V})$. Analogously to the previous chapter, $\bar{V}$ coincides with V in the votes left unchanged by the briber, but contains the new rankings in the bribed votes. Throughout this chapter, we always assume that the voters v_i are given as strict linear orders over C. If not mentioned other, we are given such an input specific to the current problem we are studying. Sometimes we rename the voters and/or candidates for practical reasons.

By default, we regard the unique-winner model. Sometimes we also consider the co-winner model to pinpoint that both winner models yield different complexities of a given problem. One reason why we focus on the unique-winner model is that Walsh and Xia [160] considered a version of the unique-winner model (although in their settings, a fixed order over candidates to break ties is given). Moreover, the unique-winner model has an advantage compared to the co-winner model: the winning probabilities of all candidates and the probability that a K-voter subelection has no unique winner sum up to one, whereas this is possibly but not necessarily the case for the co-winner model. E.g., regard the three-voter and three-candidate election (C,V) with $C = \{c_1,c_2,c_3\}$ and $V = \{v_1,v_2,v_3\}$ where the lot size is $K = 2$. Our voting rule is the Plurality rule and voter v_i votes for c_i ($1 \leq i \leq 3$). There are three 2-voter subelections and each subelection has two tying winners with one point each. In the co-winner model, the winning probabilities are $\frac{2}{3}$ for all three candidates (they sum up to two instead of one); in the unique-winner model, the winning probabilities are zero for each candidate and the probability that there is no unique winner is equal to one. Hence, the unique-winner model "normalizes" the winning probabilities to one which might be a slight advantage when studying more complex models differing from the uniform distribution assumption, the canonical assumption in our setting. In other words, the lottery selects each combination of K out of n voters with the same probability $\frac{1}{\binom{n}{K}}$.

Before we start our complexity analysis, we recall and introduce the decision and counting problems we will explore below. We start with the evaluation problem which was introduced in [160].

LotThen$\mathscr{F}$-Evaluation	
Given:	An election (C,V), a designated candidate $c \in C$, a rational number $p \in [0,1]$, and a natural number $K \leq \|V\|$.
Question:	Is the probability of c to be the LotThen$\mathscr{F}$ winner by drawing K votes strictly larger than p?

It is also interesting to compute whether there is at least one subelection in which a given candidate can win and whether the candidate wins in all subelections. This can be seen as a natural

[5] In this chapter, we assume that there is always at least one voter in V because the lot size K is always positive, too.

adaption of the possible/necessary winner problem [117] to lot-based voting rules and was also suggested as an interesting direction in [160].

LotThen$\mathscr{F}$-Possible/Necessary Winner	
Given:	An election (C,V), a designated candidate $c \in C$, and a natural number $K \leq \|V\|$.
Question:	Is c the LotThen$\mathscr{F}$ winner of at least one/of every K-voter subelection?

We point out that possible winner can be regarded as a form of electoral control where the chair can fix a K-voter subelection in advance, i.e., the chair manipulates the lottery itself and replaces the actually random sampling of the lottery.

In contrast to the evaluation problem which asks only whether the probability is greater than a given value, it can be desirable to compute the *number* of subelections in which the candidate wins. We define this counting problem as follows:

#LotThen$\mathscr{F}$	
Given:	An election (C,V), a designated candidate $c \in C$, and a natural number $K \leq \|V\|$.
Question:	How many K-voter subelections do exist with the unique winner c?

Finally, we regard two variants of bribery, namely possible and necessary bribery introduced by us in the context of lot-based voting. Note that in a lot-based setting we cannot work with the standard bribery definition where in a given election the briber is allowed to change some voters' votes in order to make his favorite candidate win the election [79]. In contrast, in lot-based elections we can ask whether there exists a K-voter subelection after the bribery where the briber reaches his goal (possible bribery) or whether the briber reaches his goal in all K-voter subelections (necessary bribery).

LotThen$\mathscr{F}$-Possible/Necessary Bribery	
Given:	An election (C,V), a designated candidate $c \in C$, a bribery limit $\ell \in \mathbb{N}_0$ (with $\ell \leq \|V\|$), and a natural number $K \leq \|V\|$.
Question:	Is it possible to change up to ℓ votes such that c is the LotThen$\mathscr{F}$ winner of at least one/every subelection of K voters?

Analogously to possible winner, possible bribery can be seen as a combination of bribery/control where the chair may completely change some voters' votes and decide which voters are drawn by the lottery.

4.3 Results

Our results are summarized and juxtaposed with results from the literature in Table 4.1. Analogously to the previous chapters, we say that some problems are NP-complete or # P-complete; however, we only prove hardness. Note that membership in NP or # P can be shown via simple guess&check algorithms.

	NW	PW	#	Eval.	NB	PB
LotThenPlurality	**P**	P [58]	**FP**	**P**	**P**	**P**
LotThen2-Approval	**P**	P [58]	**#PC**		**P**	**P**
LotThen$k$-Approval ($k \geq 3$)	**P**	NPC [58]	*#PC*	*NPC*	*NPC*	*NPC*
LotThenVeto	**P**	P [58]	**FP**	**P**	**P**	**P**
LotThen2-Veto	**P**	**P**	**#PC**		**P**	**P**
LotThen3-Veto	**P**	**NPC**	**#PC**	**NPC**	**NPC**	**NPC**
LotThen$k$-Veto ($k \geq 4$)	**P**	**NPC**	**#PC**	**NPC**	*NPC*	*NPC*

Table 4.1: Complexity results for LOTTHENk-Approval and LOTTHENk-Veto under the unique-winner model. Results in boldface are new. Results in italic follow from [119] or [58] via reductions. Key: "NW" stands for "necessary winner", "PW" means "possible winner", "#" stands for "counting complexity", "Eval." stands for "evaluation", "NB" means "necessary bribery", "PB" stands for "possible bribery", "#PC" stands for "# P-complete", and "NPC" means "NP-complete". P and FP indicate membership results.

4.3.1 Possible and Necessary Winner

Our first result says that the necessary winner problem for lot-based voting rules is easy even for the whole family of scoring rules.

Theorem 4.1. LOTTHEN$\mathscr{F}$-NECESSARY WINNER *is in* P *for every scoring rule* $\mathscr{F}$.

Proof. Our algorithm checks for every candidate c_j ($j \in [m-1]$) whether c_j can beat or tie with c for at least one subelection with K voters. To do so, we first sort the voters v_i in decreasing order according to the score differences $score_{(C,\{v_i\})}(c_j) - score_{(C,\{v_i\})}(c)$. This can be done in polynomial time. [6] Hence, for each c_j, we assume that (after sorting) the voters in V are sorted in decreasing order according to these score differences. In other words, v_1 and v_n maximize and minimize this score difference, respectively. Note that all voters might yield the same difference.

We accept if and only if it holds $\Delta(c_j) := \sum_{i=1}^{K}(score_{(C,\{v_i\})}(c_j) - score_{(C,\{v_i\})}(c)) < 0$ for each $j \in \{1,\ldots,m-1\}$ as then and only then for every K-voter subelection c has a higher score than every other candidate c_j. Observe that for each c_j, the first K score differences lead to a maximum $\Delta(c_j)$. Hence, if there is any chance for c_j to beat or tie with c in a subelection of K voters, these K voters ensure this.

Note that our problem is in P as we can first compute the difference $score_{(C,\{v_i\})}(c_j) - score_{(C,\{v_i\})}(c)$ in every vote, sort the values decreasingly, and calculate the sum of the first K values. This is done for at most $m-1$ candidates. ❑

Now let us focus on possible winner. In [58], the authors have already studied the possible winner problem for k-Approval and for the Veto voting rule in lot-based elections. [7] More precisely, they considered a variant of the unique-winner model with ties broken in a predetermined

[6] The "naive" approach (already existing in literature) for sorting a list of n numbers is as follows. Go through the whole list, move the largest element at the end of the list, leave the ordering of the remaining numbers unchanged. Go a second time through the list, move again the largest element among the first $n-1$ numbers at the end of this list with $n-1$ numbers. Apply this subroutine n times. Observe that this naive approach requires $O(n^2)$ running time.

[7] These results can be regarded as a byproduct of their work since they actually studied the so-called k-DELEGATION problem of which possible winner is a special case with $k = 1$. They did not use the term "possible winner" to denote this special case $k = 1$.

order. To tailor these results to our unique-winner model, for k-Approval ($k \leq 2$) and Veto we may also exploit that the more general problem possible bribery is polynomial-time decidable (cf. Theorem 4.27, 4.28, and 4.30). For 3-Approval, the proof of Theorem 6.1 in [58] holds for a unique-winner variant with ties broken in favor of the distinguished candidate (called z in their proof). Hence, in fact, the authors therein show the result for the co-winner case defined as in Section 2.3, for the tie-breaking rule selects c in doubt. To adapt their proof to "our" unique-winner model, we adjust their construction as follows.

Theorem 4.2. LOTTHENk-APPROVAL-POSSIBLE WINNER *is* NP*-complete for* $k \geq 3$.

Proof Sketch. We first provide the construction for $k = 3$ and argue later how to adjust it for $k > 3$. We renounce of a detailed proof since our construction is nearly identical to the one used in the proof of Theorem 6.1 in [58] and the argument of correctness works almost identically.

Given an instance $(B, \mathscr{S})$ of X3C with $B = \{b_1, \ldots, b_{3m}\}$, $\mathscr{S} = \{S_1, \ldots, S_n\}$, $S_i \subset B$, $|S_i| = 3$ for each i, $1 \leq i \leq n$, and $n \geq m$ (otherwise, a cover does not exist), we construct the following instance of LOTTHEN3-APPROVAL-POSSIBLE WINNER. Let $C = \{c, d_1, \ldots, d_4\} \dot{\cup} B$ denote the set of candidates, c be the distinguished candidate, and $K = m + 2$. Moreover, there are $n + 2$ voters in V defined as follows:

- v_i approves of the candidates in S_i ($1 \leq i \leq n$).
- Voter v_{n+1} approves of c, d_1, and d_2.
- Voter v_{n+2} approves of c, d_3, and d_4.

In order that c is a unique winner in a K-voter subelection, the lottery must select the lowermost two voters v_{n+1} and v_{n+2} and m voters in $\{v_1, \ldots, v_n\}$. The only way to ensure that c is a unique winner in such a subelection is that the lottery picks each $b_j \in B$ at most once. Since m voters from $\{v_1, \ldots, v_n\}$ approve of $3m$ candidates in total and B includes $3m$ candidates, this is only possible if an exact cover exists. In case an exact cover exists, the lottery makes c the unique winner via selecting the m cover corresponding voters and v_{n+1} and v_{n+2}.

For $k > 3$, we additionally introduce $(n+2)(k-3)$ dummy candidates each of whom is approved by exactly one voter in V.

In the following, we complete the picture by solving the possible winner problem for LotThen$k$-Veto, for all $k \geq 2$.

Theorem 4.3. LOTTHEN2-VETO-POSSIBLE WINNER *is in* P.

Proof. We omit the proof because Theorem 4.17 and 4.31 imply that the more general problem LOTTHEN2-VETO-POSSIBLE BRIBERY is in P. ❑

For 3-Veto, the complexity of the possible winner problem differs in the unique-winner and co-winner models. This is yet another case in the list of voting problems where ties matter (and again the voting rule is 3-Veto). Recall that ties matter for the "classical" possible winner problem (Theorem 3.11 and 3.12 in Section 3.4) as well.

First, we show that possible winner for 3-Veto is hard under the unique-winner model.

Theorem 4.4. LOTTHENk-VETO-POSSIBLE WINNER *is* NP*-complete for* $k \geq 3$ *under the unique-winner model.*

Proof. The proof is for 3-Veto. At the end of the proof, we will show how to adapt the proof for $k \geq 4$. We show hardness by a reduction from 3DM. Based on a 3DM instance (X,Y,Z,E) with $|X| = |Y| = |Z| = n$, $E = \{e_1,\ldots,e_m\}$ (where $E \subseteq X \times Y \times Z$), and $m \geq n$ (otherwise, a three-dimensional matching does not exist), we construct an election (C,V), where $C = \{c\} \dot\cup X \dot\cup Y \dot\cup Z$ is the set of candidates, c is the designated candidate, and $V = \{v_1,\ldots,v_m\}$ is the set of voters, where each v_i, $1 \leq i \leq m$, vetoes exactly the candidates in e_i. Furthermore, let n be the lot size. Note that $\mathit{vetoes}(c) = 0$ in the overall election. Thus, the distinguished candidate has zero vetoes in every n-voter subelection.

We claim that c is a LotThen3-Veto possible winner if and only if there is a three-dimensional matching $E' \subseteq E$.

($\Rightarrow$): Assume that c is a LotThen3-Veto possible winner. This means that every $a \in X \cup Y \cup Z$ is vetoed at least once in an n-voter subelection. As the lottery distributes exactly $3n$ vetoes by selecting n voters $v_{i_1},\ldots,v_{i_n}$ at random (where $1 \leq i_1 < \ldots < i_n \leq m$) and since there are exactly $3n$ candidates in $X \cup Y \cup Z$, the corresponding sets $e_{i_1},\ldots,e_{i_n}$ have to form a three-dimensional matching of $X \cup Y \cup Z$.

($\Leftarrow$): Assume that a three-dimensional matching $E' = \{e_{i_1},\ldots,e_{i_n}\}$ exists. Consider the n-voter subelection $(C,\{v_{i_1},\ldots,v_{i_n}\})$. In this subelection, each $a \in X \cup Y \cup Z$ receives exactly one veto. Thus, being the only candidate with zero vetoes in the resulting n-voter subelection, c is the unique winner in this n-voter subelection and thus a LotThen3-Veto possible winner.

For $k \geq 4$, we extend the construction for 3-Veto by introducing $k-3$ dummy candidates who are vetoed in every vote. ❑

The following theorem says that the same problem as in the previous theorem is solvable in polynomial time under the co-winner model.

Theorem 4.5. LOTTHEN3-VETO-POSSIBLE WINNER *is in* P *in the co-winner model.*

Proof. Again, we refer to the corresponding result for possible bribery. According to Theorem 4.17 and 4.33, LOTTHEN3-VETO-POSSIBLE BRIBERY is easy in the co-winner model. Since possible winner is a special case of possible bribery, membership in P follows. ❑

Ties matter for LOTTHEN3-VETO-POSSIBLE WINNER. The following theorem says that for $k \geq 4$, LOTTHENk-VETO-POSSIBLE WINNER is NP-complete under the co-winner model as well. The proof uses a different construction to the proof sketch in [119] pointing out that the closely related problem k-VETO-CCDV is NP-complete for $k \geq 4$. To complete the picture, we next present the full proof, tailored to the possible winner problem for lot-based voting rules.

Theorem 4.6. LOTTHENk-VETO-POSSIBLE WINNER *is* NP*-complete for every* $k \geq 4$ *in the co-winner model.*

Proof. The proof is for 4-Veto. At the end of the proof, we will show how to adapt the proof for $k \geq 5$.

We show hardness by a reduction from X3C. Our X3C instance $(B,\mathscr{S})$ is defined by $B = \{b_1,\ldots,b_{3m}\}$, $\mathscr{S} = \{S_1,\ldots,S_n\}$, $S_i \subset B$, and $|S_i| = 3$ ($i = 1,\ldots,n$). W.l.o.g., we let $n \geq m$ (otherwise,

a cover does not exist) and $m \geq 2$.[8] According to $(B, \mathscr{S})$, we construct an election (C,V), where $C = \{c, d_1, d_2, d_3\} \dot{\cup} B$ is the set of candidates, and V is the set of voters with the following $n+3m(m-1)$ voters:

1. For each i, $1 \leq i \leq n$, there is a voter v_i who vetoes c and the candidates in S_i and
2. for each j, $1 \leq j \leq 3m$, there are $m-1$ voters who veto b_j and the dummy candidates d_1, d_2, and d_3.

Furthermore, let $K = m + 3m(m-1)$ be the number of voters drawn by the lottery. Note that $vetoes(c) = n$, $vetoes(d_j) = 3m(m-1)$ for each j, $1 \leq j \leq 3$, and $vetoes(b_j) = l_j + (m-1)$, for each j, $1 \leq j \leq 3m$, where $l_j = |\{S_i \in \mathscr{S} : b_j \in S_i\}|$.

We claim that c is a LotThen4-Veto possible winner if and only if B has an exact cover.

($\Rightarrow$): Assume that c is a possible winner, hence there exists a K-voter subelection where c is a winner. Since there are only $3m(m-1)$ voters not vetoing c, and $K = m + 3m(m-1)$, the distinguished candidate c receives at least m vetoes in this subelection. First we show that in a subelection of which c is a winner, c has to get exactly m vetoes. Then we argue that an exact cover of B must exist. Assume that c receives $m+t$ vetoes. This means that $m+t$ voters of the first group and $3m(m-1)-t$ voters of the second group are drawn by the lottery. Consequently, the candidates in B receive a total of $3(m+t)+1(3m(m-1)-t)$ vetoes in such K-voter subelections. These vetoes have to be distributed among all $b_j \in B$ such that each of the $3m$ candidates in B receives at least $m+t$ vetoes. Thus, the following inequality has to hold:

$$3(m+t)+1(3m(m-1)-t) \geq 3m(m+t).$$

We can simplify this inequality to:

$$3mt \leq 2t.$$

Since t is a nonnegative integer and m is positive due to our assumptions from above, $t = 0$ is the only admissible solution to this inequality.

Therefore, c being a possible winner implies that only m voters drawn by the lottery veto c and all voters of the second group belong to this K-voter subelection.

As c has m vetoes and each b_j is vetoed by $m-1$ voters in the second voter group, $3m$ candidates in B require (at least) $3m$ additional vetoes from voters in the first group, drawn by the lottery. This is only possible when the lottery can pick m voters $v_{i_1}, \ldots, v_{i_m}$ from the first group such that the sets $S_{i_1}, \ldots, S_{i_m}$ exactly cover B.

($\Leftarrow$): Assume that B has an exact cover. Drawing the corresponding m voters from the first voter group and all voters from the second voter group makes c a winner of this subelection since $vetoes(c) = vetoes(b_j) = m$ for each j, $1 \leq j \leq 3m$, and $vetoes(d_j) = K - m = 3m(m-1)$ ($1 \leq j \leq 3$). (In particular, from $m > 1$ it follows that c has at most as many vetoes as the d_j; for $m = 1$, the d_j would have fewer vetoes than c.)

To adapt the proof for k-veto with $k \geq 5$, it suffices to introduce $k-4$ dummy candidates being vetoed in every vote. ❑

[8] This additional condition will turn out to be essential as otherwise the dummy candidates beat c.

Thus, we have achieved a complexity dichotomy for k-Veto under the unique-winner model and could show that LOTTHENk-Veto-POSSIBLE WINNER is easy for $k \leq 2$ and hard for $k \geq 3$.

Our next theorem states that our results still hold when we drop the assumption that K voters are uniformly drawn at random and relax this assumption by allowing that each combination of K voters is drawn with an individual positive probability.

Theorem 4.7. *All results for possible and necessary winner still hold if each combination of K voters is drawn by the lottery with arbitrary positive probability.*

Proof. Verify that all proofs for necessary winner and possible winner only care about whether c is the winner for at least one or all subelections with K voters. The actual probability weights are irrelevant. ❑

4.3.2 Counting Problems

In this section, we analyze the computational complexity of #LOTTHEN$\mathscr{F}$ for the voting rules studied above. Firstly, these results allow to compare the complexity of the decision problems with the counting problems. Secondly, the #LOTTHEN$\mathscr{F}$ problem can also be used to compute the probability of being a winning candidate. (Notice that this is in contrast to the evaluation problem where we only check whether the probability is larger than a specified bound.)

Theorem 4.8. #LOTTHENPLURALITY *is in* FP.

Proof. For this result we use the relation to constructive control by adding voters (CCAV) and build upon a result from the literature (Theorem 6 in [164]) which states that the counting version of CCAV for Plurality is in FP. In a nutshell, in CCAV one can add up to ℓ voters to the given election to make a designated candidate win. This is in contrast to lot-based voting rules where we have a lot-size of exactly $K = \ell$. To compute #LOTTHENPLURALITY, it suffices to construct two CCAV instances: the first one with candidate set C, empty registered voter set, unregistered voter set V, and with K voters allowed to be added from V to the election; the second one is the same, but permits only $K-1$ voters to be added.

Let N be the solution count for the instance with K and N' for the instance with $K-1$, respectively. We compute the difference $N-N'$ which gives us the number of ways to add exactly K voters from V to the empty election such that c is the unique winner. This is equivalent to the lot-based possible winner problem for Plurality and hence #LOTTHENPLURALITY is in FP. ❑

Our next result reveals that we can check in polynomial time whether c is a possible winner, but counting the number of K-voter subelections with c as the sole winner is hard. The proof uses a similar construction to the proof of Theorem 10 in [164] showing that k-APPROVAL-#CCAV is # P-complete for $k \geq 2$.

Theorem 4.9. #LOTTHENk-APPROVAL *is* # P*-complete for $k \geq 2$.*

Proof. In a first step, we show hardness for $k = 2$ and argue later how to adapt the proof for larger values. We reduce from #PERFECT MATCHINGS which is well-known to be # P-complete [157]. Given an instance $(X, Y, \mathscr{E})$ with $|X| = |Y| = n$ and $\mathscr{E} = \{e_1, \ldots, e_m\}$ (each edge connects a vertex in

X with a vertex in Y), and $m \geq n$ (otherwise, a matching is impossible) of #PERFECT MATCHINGS, we construct an instance of #LOTTHEN2-APPROVAL as follows.

Let $C = \{c, d_1, d_2\} \dot\cup X \dot\cup Y$, where c is the designated candidate. The voters in V are defined as follows.

1. For each $e_i = \{x^i, y^i\} \in \mathscr{E}$ ($i \in \{1, \ldots, m\}$, $x^i \in X$, $y^i \in Y$), there is a voter ("edge voter") v_i approving of x^i and y^i.
2. Besides, there are two additional voters v_{m+1} and v_{m+2} approving of the candidate pairs $\{c, d_1\}$ and $\{c, d_2\}$, respectively.

Finally, let $K = n + 2$.

Observe that if c is drawn at most once, c cannot be the unique winner as there is at least one candidate approved of by an edge voter who would be tied with c or beat c. Hence, a subelection with c as the winner must have v_{m+1} and v_{m+2} among its voters. Now n remaining voters give $2n$ points to $2n$ non-distinguished candidates. Consequently, the only way for c to win with two approvals is the existence of a perfect matching since then every non-distinguished candidate is drawn exactly once and none of the candidates in $X \cup Y$ scores at least twice in the drawn votes. The dummy candidates d_1 and d_2 have only a score of one each and therefore lose against c if v_{m+1} and v_{m+2} are picked by the lottery. As the number of winning subelections equals the number of perfect matchings, hardness for our problem follows immediately.

For higher values of k we simply introduce $(m+2)(k-2)$ dummy candidates who are approved of precisely once in total. ❑

Before we show our result for the Veto rule, we first prove an FP result left open in [164]. Based on this easiness result, we will be able to settle our FP result for #LOTTHENVETO in the subsequent theorem.

Theorem 4.10. VETO-#CCAV *is in* FP.

Proof. To prove our result, we use a recursion based on a similar approach to the one in the proof of Theorem 6 in [164] (showing that PLURALITY-#CCAV is in FP).

Our input is an election $(C, V \dot\cup W)$ with $m > 1$ candidates in candidate set $C = \{c, c_1, \ldots, c_{m-1}\}$, n_V voters in registered voter set V, n_W voters in unregistered voter set W, distinguished candidate c, and a nonnegative integer $\ell \leq n_W$ denoting the number of voters the chair may add from W.

First let V_d and W_d contain all voters in V and W vetoing candidate $d \in C$, respectively. Our goal is to count the number of subsets W' added from W such that $|W'| \leq \ell$ holds and c is the unique winner in the final election $(C, V \cup W')$. In case j added voters veto c ($j \in \{0, 1, \ldots, \min(\ell, |W_c|)\}$), the chair can additionally introduce any number of voters between 0 and $\ell - j$ from $W_{c_1} \cup \ldots \cup W_{c_{m-1}}$.

Let $N'(C, V, W, c, \ell, j)$ be the number of ways that (1) the chair adds exactly j voters vetoing c from W and (2) at most $\ell - j$ other voters (i.e., from $W_{c_1} \cup \ldots \cup W_{c_{m-1}}$), and (3) c is the unique winner in the resulting election. Observe that the total number $N(C, V, W, c, \ell)$ of possibilities to add ℓ voters or less from W such that c is the unique winner in the final election can be computed the following way:

$$N(C, V, W, c, \ell) = \sum_{j=0}^{\min(|W_c|, \ell)} N'(C, V, W, c, \ell, j).$$

In the following, we show how to derive the values $N'(C,V,W,c,\ell,j)$ (for each $j \in \{0,1,\ldots,\min(\ell,|W_c|)\}$) via recursion. One idea is that we first compute this value restricted to one candidate c_1 and for any possible number of added vetoes for non-distinguished candidates. Based on these values, we can compute these numbers embracing the candidates c_1 and c_2 (that is, c must beat both c_1 and c_2), and step by step, we can amplify the number of candidates restricting ourselves to until we incorporate all candidates other than c. Accordingly, for a given number j of added voters vetoing c, we recursively determine the numbers of ways to add at most $\ell - j$ voters vetoing the other candidates such that $c_1,\ldots,c_{m-1}$ have more than $vetoes_{(C,V)}(c) + j$ vetoes in the final election.

Formally, we let a_{ti}^j denote the number of subsets $W'' \subseteq W_{c_1} \cup \ldots \cup W_{c_i}$ added from W such that $|W''| = t$ holds and every c_h ($1 \leq h \leq i$) has more than $|V_c| + j$ vetoes in total (including vetoes from voters in V and from voters added from W). Note that it holds $j \in \{0,1,\ldots,\min(|W_c|,\ell)\}$, $t \in \{0,1,\ldots,\ell - j\}$, and $i \in \{1,\ldots,m-1\}$. j denotes the index representing the additional number of vetoes for c from voters added from W, i is the candidate index (i.e., the number of non-distinguished candidates to whom we restrict ourselves), and t denotes the index representing the number of voters added from $W_{c_1} \cup \ldots \cup W_{c_i}$.

In other words, when computing a_{ti}^j, we actually calculate the number of ways to add exactly t voters from W vetoing the first i non-distinguished candidates such that c beats each of them, by implicitly assuming that j further added voters veto c, that is, c is the unique winner for the final election restricted to the candidate subset $\{c,c_1,\ldots,c_i\}$ even though we increase the original veto number of c by j. Note that in this context we do not consider the number of ways to select j vetoes for c from $|W_c|$ voters in W_c. We compute the number of ways for c to get j vetoes from voters in W separately at the end of the proof.

As aforementioned, since c has $|V_c| + j$ vetoes in the final election, every c_h ($1 \leq h \leq i$) must have at least $|V_c| + j + 1$ vetoes in the final election. We apply the following recursive algorithm to our problem. For each $j \in \{0,\ldots,\min(|W_c|,\ell)\}$, $t \in \{0,\ldots,\ell - j\}$, and $i \in \{1,\ldots,m-1\}$, we compute the values a_{ti}^j as follows. (In the Appendix, we will provide a numerical example in order to illustrate this algorithm.)

- If $t = 0$, $i \geq 1$, compute $a_{ti}^j = \begin{cases} 1, & \text{if } |V_c| + j < \min(|V_{c_1}|,\ldots,|V_{c_i}|) \\ 0, & \text{else} \end{cases}$.

  When we add $t = 0$ voters not vetoing c, there is only one possibility to "add" zero vetoes for the first i non-distinguished candidates such that c is the winner with j additional vetoes: the case where c beats the first i candidates $c_1,\ldots,c_i$ who receive only their vetoes from voters in V and c even gets j additional vetoes.

- If $t > 0$, $i = 1$, compute $a_{t1}^j = \begin{cases} \binom{|W_{c_1}|}{t}, & \text{if } t > |V_c| + j - |V_{c_1}|, \\ 0, & \text{else} \end{cases}$.

  We can only add t vetoes for the first candidate c_1 and c has fewer vetoes than c_1 in the final election (even though j additional vetoes for c are considered) if t is large enough and there are enough vetoes for c_1 in W to add.

- In case $t > 0$ and $i > 1$, compute $a_{ti}^j = \sum_{s=\max(|V_c|+j-|V_{c_i}|+1,0)}^{\min(|W_{c_i}|,t)} \binom{|W_{c_i}|}{s} a_{t-s,i-1}^j$.

Every summand behind the sigma sign counts the number of ways to add s vetoes for the ith candidate from W such that c_i and all candidates $c_1,\ldots,c_{i-1}$ have more vetoes than c in the final election (where c has $|V_c|+j$ vetoes) and exactly $t-s$ vetoes in total for the first $i-1$ candidates (and thus exactly t vetoes for the first i non-distinguished candidates) are added from W. The lower bound of the sum describes that c_i requires at least $\max(|V_c|+j-|V_{c_i}|+1,0)$ additional vetoes from voters in W to not beat or tie with c in the final election (as c has $|V_c|+j$ vetoes in the final election; since c_i is vetoed by $|V_{c_i}|$ voters in V, c_i requires at least $\max(|V_c|+j-|V_{c_i}|+1,0)$ additional vetoes from voters added from W in order to not tie with or even beat c). Zero additional vetoes only suffice for c_i to have more vetoes than c when c_i already receives more than $|V_c|+j$ vetoes from voters in V. The upper bound of the sum ensures that the chair can add no more than t, but also no more than $|W_{c_i}|$ voters from W_{c_i}. Note that in case the upper bound is smaller than the lower bound, we obtain a zero sum. The same holds for binomial coefficients where the upper number is smaller than the lower one. Observe that there are $\binom{|W_{c_i}|}{s}$ ways to select s vetoes for c_i in W. In case the chair adds s voters vetoing c_i and t voters vetoing the first i candidates in total, there are $a^j_{t-s,i-1}$ combinations to select the remaining voters added from $W_{c_1}\cup\ldots\cup W_{c_{i-1}}$ such that the first i candidates have more vetoes than c in the final election. The sum implicitly exploits that c has $|V_c|+j$ vetoes in the final election.

Recall that a^j_{ti} counts the number of ways to add exactly $t\leq \ell-j$ voters vetoing the candidates $c_1,\ldots,c_i$ (i.e., the first i candidates, $i\leq m-1$) such that every candidate in $\{c_1,\ldots,c_i\}$ has more vetoes than c in the final election, under the assumption that j voters vetoing c are added from W. [9] Note that we compute a^j_{ti} (for the first i candidates), directly exploiting the values $a^j_{t'i'}$ ($t'=0,\ldots,t$, $i'=1,\ldots,i-1$).

One can regard the j index as the outermost loop, t as the middle loop, and the i index as the innermost loop, i.e., based on the current values for j and t, our algorithm provides the values a^j_{ti} for $i=1,\ldots,m-1$.

The number $N(C,V,W,c,\ell)$, counting all possible ways to add at most ℓ voters from W and c is the unique winner afterwards, is computed as follows:

$$N(C,V,W,c,\ell)=\sum_{j=0}^{\min(|W_c|,\ell)} N'(C,V,W,c,\ell,j)$$

$$=\sum_{j=0}^{\min(|W_c|,\ell)}\binom{|W_c|}{j}\cdot\sum_{t=0}^{\ell-j}a^j_{t,m-1}.$$

Observe that there are only polynomially many values a^j_{ti} to determine. More precisely, there are $O(\ell^2 m)$ such numbers to compute and each one in turn is easy to calculate. Thus, this sum can be computed in polynomial time and our overall counting problem is in FP. ❑

Directly exploiting the algorithm given in the proof of Theorem 4.10, we can settle the number of ways to draw K out of n voters and c is the only winner:

[9] Remind that the number of combinations of j vetoes being selected from $|W_c|$ voters in W_c do no matter here. We consider these numbers for every j when calculating the final sum at the end of the proof.

Theorem 4.11. #LOTTHENVETO *is in* FP.

Proof. Analogously to the proof of Theorem 4.8, we define two instances of VETO-#CCAV and compute $N = N(C,\emptyset,V,c,K) - N(C,\emptyset,V,c,K-1)$, where $N(C,\emptyset,V,c,K)$ is defined as the number of ways to add *at most* ℓ voters from V to an election without any voters and c is the unique winner. Likewise, subtracting two sums of problems easy to count is in FP as well. ❑

Our next result offers us another instance where the decision problem is easy, but the counting problem is hard (for $k = 2$):

Theorem 4.12. #LOTTHENk-VETO *is* # P*-complete for* $k \geq 2$.

Proof. Our proof is only for $k = 2$. At the end, we will argue how to adapt the proof to higher values of k. For 2-Veto, we reduce #PERFECT MATCHINGS to our problem. Given an instance $(X,Y,\mathscr{E})$ of #PERFECT MATCHINGS with $|X| = |Y| = n$ and $\mathscr{E} \subseteq \{\{x,y\} : x \in X, y \in Y\}$, $\mathscr{E} = \{e_1,\ldots,e_m\}$, we construct our election as follows. We are given a candidate set $C = \{c\} \dot\cup X \dot\cup Y$, where c denotes our distinguished candidate. Our voters are defined as follows. For $e_i = \{x^i,y^i\}$ ($x^i \in X, y^i \in Y, i = 1,\ldots,m$), there is a voter vetoing x^i and y^i. Finally, our lot size is $K = n$. W.l.o.g., we have $m \geq n$ as otherwise a perfect matching does not exist and our election would not be well-defined (as the lot size would be greater than the number of voters).

Notice that c has zero vetoes, no matter which K voters are drawn. However, c only wins alone if and only if every candidate in $X \cup Y$ is vetoed at least once in a given subelection of $K = n$ voters. As $2n$ vetoes go to $2n$ candidates in total, this implies that c can only be the winner if and only if the n drawn votes form a perfect matching of X and Y. As the number of winning subelections equals the number of perfect matchings and this latter number is hard to count, our counting problem is hard as well. For higher values of k, we introduce $k-2$ dummy candidates $d_1,\ldots,d_{k-2}$ who are vetoed in every vote. ❑

Thus, we obtain dichotomy results both for k-Approval and k-Veto. For both voting rules, the cases $k = 1$ are easy to count and the other cases are hard to count.

4.3.3 Evaluation

In this section we present some evaluation results for lot-based voting rules not considered by Walsh and Xia [160]. We start with a theorem which relates the possible winner, necessary winner, and the evaluation problems.

Theorem 4.13. *For every voting rule* $\mathscr{F}$ *the following holds:*

1. LOTTHEN$\mathscr{F}$-POSSIBLE WINNER $\leq_m^p$ LOTTHEN$\mathscr{F}$-EVALUATION *and*
2. LOTTHEN$\mathscr{F}$-NECESSARY WINNER $\leq_m^p$ LOTTHEN$\mathscr{F}$-EVALUATION,

where $\leq_m^p$ *means polynomial-time many-one reducibility.*

Proof.

1. The possible winner problem is a special case of the evaluation problem with $p = 0$.

2. Let n be the number of voters and K the lot size. For a candidate c, a winning probability of $p = 1 - 1/\binom{n}{K}$ under the uniform distribution means that there is exactly one K-voter subelection where c is not the unique winner. If we use this probability p in the evaluation problem, we require that c is the unique winner in all K-voter subelections and the underlying problem is actually the necessary winner problem. Hence, the necessary winner problem is a special case of the evaluation problem with $p = 1 - 1/\binom{n}{K}$.

❑

We point out that both reductions still hold when we relax the assumption of the uniform distribution and allow for general positive probabilities for every combination of K voters to be drawn by the lottery. Let p_{min} denote the smallest probability with which a combination of K voters is drawn by the lottery. Then $p = 1 - p_{min}$ is a possible probability threshold for necessary winner, while $p = p_{min}$ is a threshold for possible winner.

Our next result gives us a direct connection between easy counting problems and their corresponding evaluation problems.

Theorem 4.14. LOTTHENPLURALITY-EVALUATION *and* LOTTHENVETO-EVALUATION *are in* P.

Proof. Let $p \in [0,1]$ be a probability threshold and regard Plurality or Veto. Due to Theorem 4.8 and 4.11, the number N of K-voter subelections with unique-winner c can be easily computed. To evaluate if c is the winner with a probability greater than p, we only have to check whether $\frac{N}{\binom{n}{K}} > p$ holds. ❑

Due to Theorem 4.13 and 4.4, and the fact that LOTTHEN3-APPROVAL-POSSIBLE WINNER is NP-complete [58], hardness follows for k-Approval and k-Veto for every $k \geq 3$ in the unique-winner model.

Corollary 4.15. LOTTHENk-APPROVAL-EVALUATION *and* LOTTHENk-VETO-EVALUATION *are* NP*-complete for* $k \geq 3$ *in the unique-winner model.*

Note that in the co-winner model, we can perform this latter reduction only for k-Veto, $k \geq 4$. Yet one can show that evaluation is hard for 3-Veto too.

Theorem 4.16. LOTTHEN3-VETO-EVALUATION *is* NP*-complete in the co-winner model.*

Proof. We prove NP-hardness by a reduction from X3C. Given an X3C instance $(B, \mathscr{S})$ with $B = \{b_1, \ldots, b_{3m}\}$, $\mathscr{S} = \{S_1, \ldots, S_n\}$, $S_i \subset B$, and $|S_i| = 3$ $(i = 1, \ldots, n)$, we construct the following election.

Let $C = \{c, d, e\} \dot{\cup} B$ be the set of candidates, c the designated candidate, and let V consist of the following $n+1$ voters:

- For each i, $1 \leq i \leq n$, there is a voter v_i vetoing the candidates in S_i and
- there is one voter v_{n+1} vetoing the candidates c, d, and e.

Furthermore, let $p = \binom{n}{m+1}/\binom{n+1}{m+1}$ be the probability threshold and $K = m+1$ the lot size. W.l.o.g., we let $n \geq m$ because otherwise an exact cover does not exist.

We claim that the probability of c being a LotThen3-Veto winner is strictly larger than p if and only if B has an exact cover.

($\Rightarrow$): Assume that the probability of c being a winner is strictly larger than p. If the lottery only chooses $m+1$ voters of the first voter group, c is a LotThen3-Veto winner with zero vetoes (recall, in the co-winner model a candidate with zero vetoes is always a winner). There are $\binom{n}{m+1}$ different ways to choose $m+1$ voters from the first voter group. Hence, for exceeding the winning probability p, there has to be an $(m+1)$-voter subelection containing voter v_{n+1} so that c is a winner. This means that $\mathit{vetoes}(c) = \mathit{vetoes}(d) = \mathit{vetoes}(e) = 1$ in this subelection. The distinguished candidate c can only be a winner if the lottery picks each $b_j \in B$ at least once. As B contains $3m$ candidates and the lottery picks exactly m votes from voter group one assigning $3m$ vetoes to candidates in B in total, the existence of an exact cover of B follows.

($\Leftarrow$): Assume that B has an exact cover. Again, the lottery can pick in $\binom{n}{m+1}$ different ways $m+1$ voters from the first voter group. In all these subelections c is a winner with zero vetoes. Additionally, picking the exact cover corresponding voters from voter group one and voter v_{n+1} is a subelection where c is a winner (each candidate in the election has exactly one veto). Thus, the winning probability of c is at least $(1+\binom{n}{m+1})/\binom{n+1}{m+1} > p$. ❑

Note that the problem is open for 2-Approval and 2-Veto, under both winner models. Although we know that counting the number of winning subelections for c is hard for both voting rules, and even though we implicitly require these numbers to compare them with the probability thresholds p, we cannot formally derive hardness of the evaluations problems from hard counting problems. Nevertheless, our intuition is that both problems are hard.

4.3.4 Bribery

In this section, we investigate two different types of bribery for lot-based voting rules, namely necessary bribery and possible bribery. Before arriving at the actual complexity analysis, we provide the following reductions:

Theorem 4.17. *For every voting rule $\mathcal{F}$, the following holds:*

1. LotThen$\mathcal{F}$-Necessary Winner $\leq_m^p$ LotThen$\mathcal{F}$-Necessary Bribery,
2. LotThen$\mathcal{F}$-Possible Winner $\leq_m^p$ LotThen$\mathcal{F}$-Possible Bribery,
3. $\mathcal{F}$-Bribery $\leq_m^p$ LotThen$\mathcal{F}$-Necessary Bribery,
4. $\mathcal{F}$-Bribery $\leq_m^p$ LotThen$\mathcal{F}$-Possible Bribery

where $\leq_m^p$ means polynomial-time many-one reducibility.

Proof. The first two relations follow since under a bribery limit $\ell = 0$ necessary and possible bribery coincide with necessary and possible winner, respectively. The third (fourth) relation holds as LotThen$\mathcal{F}$-Necessary Bribery (LotThen$\mathcal{F}$-Possible Bribery) instances with $K = n$ (i.e., the lottery draws all voters in V) coincide with standard bribery. ❑

Before we settle the first complexity result, we provide an auxiliary result.

Lemma 4.18. *Let $\mathscr{F} = k$-Approval. Suppose that the distinguished candidate c has at least $\bar{s}$ points in every K-voter subelection and there is at least one K-voter subelection for which c has exactly $\bar{s}$ points. Then (under the unique-winner model) c is a necessary winner if and only if each other candidate c_j has at most $\bar{s}-1$ points in total.*

Proof. The minimum guaranteed score $\bar{s}$ for c in any K-voter subelection is uniquely determined. Assuming that c has $score(c)$ points in a given election and the lottery picks as many voters as possible disapproving of c, it holds $\bar{s} = \max(0, score(c) - n + K)$. This finding can be interpreted as follows. In case c has more than $score(c) - n + K$ approvals, the lottery can ignore $n - K$ voters approving of c and selects exactly $\bar{s} = score(c) - n + K > 0$ voters approving of c. In case c has at most $n - K$ points in total, the lottery ignores all c-voters and therefore no drawn voter approves of c.

Next, we show that c is a necessary winner with minimum guaranteed score $\bar{s}$ if and only if for each c_j it holds $score(c_j) \leq \bar{s} - 1$.

($\Leftarrow$): As c_j has at most $\bar{s}-1$ points in total, this holds for any subelection with K voters. Since c has $\bar{s}$ or more points in any K-voter subelection, c is the only winner no matter which K voters are drawn by the lottery.

($\Rightarrow$): Assume for contradiction that c is the unique winner for every K-voter subelection, but there is some c_j with $score(c_j) \geq \bar{s}$. We distinguish the following cases:

- $score(c_j) \geq \max(K, \bar{s})$. In this case, we can construct a K-voter subelection where K voters approving of c_j are drawn. Hence, c is not the only winner in this subelection and thus no necessary winner.

- $\bar{s} \leq score(c_j) < K$. Observe that the lottery cannot draw K voters approving of c_j, but we may assume that it selects as many c_j-voters as possible. We construct the following subelection for which c is not a unique winner.

  1. Pick all $K - \bar{s}$ voters not approving of c. These voters exist as c has only $\bar{s}$ points in the worst-case.
  2. Draw all $\min(\bar{s}, score(c, c_j))$ voters approving of c and c_j.
  3. Select $\bar{s} - \min(\bar{s}, score(c, c_j))$ voters approving of c, but not c_j.

  If $\min(\bar{s}, score(c, c_j)) = \bar{s}$, the lottery draws $\bar{s}$ voters approving of c and c_j. Since c has exactly $\bar{s}$ points and c_j at least $\bar{s}$ points in this subelection, c is not the unique winner. If $score(c, c_j) < \bar{s}$, the lottery selects all $score(c, c_j)$ voters approving of c and c_j and all voters approving of c_j, but not c. Hence, c_j has at least $\bar{s}$ points in this subelection and c has exactly $\bar{s}$ points.

Observe that each considered subcase results in a contradiction. ❑

We obtain an analogous result for k-Veto by exploiting that a voter vetoes a given candidate if and only the voter disapproves this candidate. Thus, we can likewise determine $\bar{v} \in \{0, \ldots, K\}$ as the largest possible veto number in any K-voter subelection. Observe that it holds $\bar{s} + \bar{v} = K$.

We start with necessary bribery asking whether the briber can make c the winner for every K-voter subelection.

Theorem 4.19. LOTTHENPLURALITY-NECESSARY BRIBERY *is in* P.

Proof. First, if $score(c)+\ell > n-\frac{K}{2}$, we accept because the briber can make c reach more than fifty percent of all votes in each K-voter subelection: c gains at least $\lfloor\frac{K}{2}\rfloor+1$ points in any such subelection and consequently, each other candidate has at most $K-\lfloor\frac{K}{2}\rfloor-1<\lfloor\frac{K}{2}\rfloor+1$ points.

For $score(c)+\ell \leq n-K$, there are selections of K voters so that c does not score at all and, in particular, c is never a unique winner with zero points in any election (in fact, c would not even be a co-winner). Thus, we reject.

It remains to argue for $n-K < score(c)+\ell \leq n-\frac{K}{2}$. In this case, c reaches a positive score in every subelection of K voters and can achieve at most a weak majority of points (that is, half of all points) in the worst case. The briber can make c reach at least a score of $\bar{s} := score(c)+\ell-n+K \in \{1,\ldots,\lfloor\frac{K}{2}\rfloor\}$ in any subelection. Notice that after the bribery, the non-distinguished candidates have a total score of $K-\bar{s}$, but before the bribery, they have $K-\bar{s}+\ell$ points altogether. As the guaranteed worst-case score of c is $\bar{s}$, the briber must make sure that the other candidates have a score of at most $\bar{s}-1$ each since in c's worst-case, the lottery draws all remaining voters not supporting c (cf. Lemma 4.18). Thus, the briber must ensure that no candidate other than c has more than $\bar{s}-1$ points in the final election. Or, in other words, ℓ bribes must suffice to keep the score of every non-distinguished candidate below $\bar{s}$ which leads to the following condition equivalent to a successful bribery:

$$\ell \geq \sum_{j=1}^{m-1} \max(0, score(c_j)-\bar{s}+1).$$

Note that this problem is surely in P since it is easy to compute the scores for all candidates and processing the different subcases is easy, too. ❑

In practice, the briber applies a greedy algorithm as in the case of bribery under full information [79] according to which he step by step bribes a voter favoring the currently best non-distinguished candidate. We now turn to 2-Approval.

Theorem 4.20. LOTTHEN2-APPROVAL-NECESSARY BRIBERY *is in* P.

Proof. Armed with Lemma 4.18, we know that the briber must reach $score_{(C,\bar{V})}(c_j) < \bar{s}$ for every c_j. We distinguish the following cases:

- If $score(c)+\ell \leq n-K+1$, we reject because there are subelections for which c is approved by at most one voter. If no voter approves of c (this is possible when $score(c)+\ell \leq n-K$ holds), c cannot be the winner. The same holds for $score(c)+\ell = n-K+1$, that is, there are K-voter subelections for which exactly one voter approves of c. Since there is another candidate approved aside from c, this candidate beats or ties with c in this subelection.

- For $n-K+2 \leq score(c)+\ell \leq n$, the briber can make c achieve at least $\bar{s} = score(c)+\ell-n+K \in \{2,\ldots,K\}$ points in any K-voter subelection. (Note that $\bar{s} \geq 2$ implies that $K \geq 2$.) We may assume that all voters disapproving of c after the bribery are picked by the lottery—and exactly $\bar{s}$ voters approving of c after the bribery join them. In order to make c the winner for each such subelection, we need to ensure that each c_j is approved by at most $\bar{s}-1$ voters in the end. Observe that a rational briber bribes only voters originally not approving of c

because this maximizes the relative gain of c against all other candidates. Note that in case $score(c,c_j) \geq \bar{s}$ for some c_j, we immediately reject since there are subelections for which all $\bar{s}$ voters approving of c and c_j are picked by the lottery and hence c at most ties with c_j for the victory. Bribing some voters approving of c and c_j, in order to get rid off some common approval(s) for c and c_j, fails in this case, too, as this decreases the guaranteed score of c to $s' < \bar{s}$ and again it is possible that all remaining voters approving of c and c_j are drawn by the lottery. Henceforth, we assume that $score(c,c_j) < \bar{s}$ for each j, $1 \leq j \leq m-1$.

Similarly to the proof of Theorem 3.19, we reduce our problem to a b-edge matching problem defined as follows. We are given an undirected multigraph $G = (\mathscr{V}, \mathscr{E})$ with vertices $\mathscr{V} = C \setminus \{c\}$. Each voter approving of c_i and c_j ($1 \leq i,j \leq m-1$, $i \neq j$) yields an edge $\{c_i,c_j\}$ in $\mathscr{E}$. As upper capacities, we are given $b(c_j) = \bar{s} - 1 - score(c,c_j)$. [10]

Analogously to the proof of Theorem 3.19, it follows that the briber can make c a necessary unique winner with ℓ bribes if and only if there is a maximum b-edge matching with at least $n - score(c) - \ell$ edges and the following inequality holds:

$$\ell \leq (\sum_{j=1}^{m-1} (\bar{s} - score(c,c_j) - 1)) - 2(n - score(c) - \ell).$$

The main differences are as follows. Firstly, in opposition to the proof of Theorem 3.19, each vote in this proof is complete. (Nevertheless, the other proof also holds for $V_0 = V_1 = \emptyset$ and then the argument of correctness is almost identical to the one in this proof.) Secondly, the existence of a matching with $n - score(c) - \ell$ edges is a necessary condition (instead of $|V_2^{\neg c}| - \ell'$ edges as in the other proof). If such a matching does not exist, the capacity constraints implicate that we can construct a K-voter subelection of the final election such that c has only $\bar{s}$ points and some c_j has $\bar{s}$ or more points and thus beats or ties with c whenever we consider as many voters as possible approving of c_j but not c, and when we account for up to $\bar{s}$ voters approving of both c and c_j, cf. the reasoning in the proof of Lemma 4.18. Thirdly, the inequality must hold as otherwise some c_j has $\bar{s}$ or more points when we account for ℓ approvals the bribed voters assign to non-distinguished candidates after the bribery. Last but not least, the capacity constraints and the inequality are adapted to the unique-winner model.

- $score(c) + \ell > n$. The briber can make c reach a full approval score after the bribery. Consequently, c is approved by each voter in every subelection of K voters after the bribery. Observe that it must hold $score_{(C,\bar{V})}(c,c_j) < K = \bar{s}$ for each c_j. Otherwise, the lottery may pick K voters approving of c and the same c_j, and c is not the unique winner. Hence, the briber must ensure with highest priority that "overhanging" points above $K-1$ must be taken away from each c_j. Accordingly, we propose the following greedy algorithm:

  - Bribe all $n - score(c)$ voters disapproving of c in the original election. This is essential in order to ensure that c has a full score of K points in every K-voter subelection.
  - Greedily bribe further $\ell - n + score(c) (> 0)$ voters so that in each step the briber bribes a voter in V^c (=the voters who approve of c in the initial election (C,V)) approving

[10] As $\bar{s} > score(c,c_j)$ holds for each c_j, due to our assumption from above, the capacity constraints are always nonnegative.

of c_j with $c_j \in \operatorname{argmax}_{1\leq i\leq m-1} score_{(C,V^c)}(c_i)$ defined as the set of non-distinguished candidates with the presently largest approval score. After each step, we decrease $score_{(C,V^c)}(c_j)$ by one and go to the next step until the bribery limit is reached.

– After the bribery, each bribed voter approves of c and ℓ approvals for other candidates must still be allocated. Greedily give the current approval to a c_j with the presently smallest approval score $score_{(C,V^c)}(c_j)$. Update the scores in each step.

Observe that this procedure keeps $\max(score_{(C,V^c)}(c_1),\ldots,score_{(C,V^c)}(c_{m-1}))$ as small as possible in each step and thus—if possible—each c_j shares fewer than K common approvals with c after the bribery. Verify that c can be made a necessary winner by means of this greedy algorithm if and only if $\ell-(n-score(c))\geq\sum_{j=1}^{m-1}\max(0,score(c,c_j)-K+1)$ and $(K-1)(m-1)\geq n$ hold. The first condition states that the briber achieves by bribing $\ell-n+score(c)$ voters already approving of c before the bribery that each c_j has temporarily (i.e., not considering ℓ approvals the bribed voters assign to non-distinguished candidates after the bribery) fewer than K points. The second condition implies that there are few enough voters. Notice that in case $(K-1)(m-1)<n$, one c_j necessarily reaches K or more points.

❑

It follows from Theorem 4.17 that the necessary bribery problem for k-Approval is NP-complete for lot-based voting rules for $k\geq 3$. Although Lin [119] only provided a hardness proof for standard bribery in k-Approval ($k\geq 3$) in the co-winner model, his hardness proof can be adapted to the unique-winner model with some minor changes in the construction.

Theorem 4.21. *k-APPROVAL-BRIBERY is NP-complete for $k\geq 3$ under the unique-winner model.*

Proof. We tailor the proof in [119] to the unique-winner model as follows (the proof is also very similar to the one showing that APPROVAL-BRIBERY is NP-complete [79]). We reduce from an instance $(B,\mathscr{S})$ of X3C with $B=\{b_1,\ldots,b_{3m}\}$, $\mathscr{S}=\{S_1,\ldots,S_n\}$, $|S_i|=3$, $S_i\subset B$ ($1\leq i\leq n$), and w.l.o.g. $n\geq m$ (otherwise, a cover cannot exist). Based on such an instance, we construct an election (C,V) with $C=B\,\dot\cup\,\{c\}\,\dot\cup\,D$, where D is a set of $6mn+6m-4n+2$ dummy candidates. Moreover, V contains of $3mn+2m-n+1$ voters divided into the following groups:

1. For each i, $1\leq i\leq n$, there is a voter v_i approving of the candidates in S_i.

2. For each j, $1\leq j\leq 3m$, there are $n+1-l_j$ voters w_i approving of b_j, where $l_j:=|\{S_i\in\mathscr{S}:\ S_i\ni b_j\}|$. Each voter in this group approves of exactly one candidate in B and two dummy candidates such that no dummy candidate is approved at least twice.

3. There are $n-m+1$ voters u_i approving of c and two other dummy candidates (each of whom is approved only once in total).

Finally, the bribery limit is $\ell=m$. The scores are $score(c)=n-m+1$, $score(b_j)=n+1$ ($b_j\in B$), and $score(d)\leq 1$ for each $d\in D$. Note that there are exactly $2m$ dummy candidates not scoring, the remaining $6mn+4m-4n+2$ receive one point each. Similarly to the proofs in [119] and [79], we accept if and only if there is an exact cover. Then, and only then, the briber bribes the

voters v_i according to the cover, c has $n+1$ points, all b_j have n points, and each d has one point in the final election. The bribed voters assign m approvals to c and $2m$ approvals to those dummy candidates not scoring in the initial election.

For $k > 3$, we introduce further $|V|(k-3)$ dummy candidates approved once in total each. ❑

Theorem 4.21 implicates the following result:

Corollary 4.22. LOTTHENk-APPROVAL-NECESSARY BRIBERY *is* NP*-complete for* $k \geq 3$ *under the unique-winner model.*

Now we turn to Veto.

Theorem 4.23. LOTTHENVETO-NECESSARY BRIBERY *is in* P.

Proof. First, observe that a rational briber bribes as many voters as possible vetoing c and each of them vetoes a non-distinguished candidate after the bribery. Accordingly, c has $\bar{v} := \max(0, \mathit{vetoes}(c) - \ell)$ vetoes in the final election. Our algorithm checks whether c is always a unique winner whenever the lottery picks as many voters as possible vetoing c.

If $\bar{v} \geq K$, we immediately reject since there are K-voter subelections containing only voters vetoing c.

Henceforth, let $\bar{v} < K$. First, suppose that $\mathit{vetoes}(c) \geq \ell$. Then the briber bribes ℓ voters vetoing c and c has $\bar{v} = \mathit{vetoes}(c) - \ell \geq 0$ vetoes after the bribery. We suppose that all $\bar{v}$ voters vetoing c are drawn by the lottery. Now the question arises which c_j should get how many vetoes from the bribed voters. Observe that c_j and all other non-distinguished candidates must have at least $\bar{v}+1$ vetoes each in any subelection of K voters. This is only possible when $K \geq \bar{v} + (m-1)(\bar{v}+1)$. Otherwise, we immediately reject as the lot size K is too small. Provided that $K \geq \bar{v} + (m-1)(\bar{v}+1)$, it must hold that all candidates in $C \setminus \{c, c_j\}$ have at most $K - \bar{v} - (\bar{v}+1)$ vetoes after the bribery altogether. This, in turn, implies that c_j must have at least $n - K + \bar{v} + 1$ vetoes in total. Otherwise, the lottery can draw too many vetoes for candidates in $C \setminus \{c, c_j\}$ and c_j has at most $\bar{v}$ vetoes in a K-voter subelection where $\bar{v}$ voters veto c. [11] Hence, we accept if and only if ℓ is large enough to ensure that each c_j can be given at least $\max(0, n - K + \bar{v} + 1 - \mathit{vetoes}(c_j))$ additional vetoes in the bribed votes, i.e., our instance is a YES instance if and only if $\ell \geq \sum_{j=1}^{m-1} \max(0, n - K + \bar{v} + 1 - \mathit{vetoes}(c_j))$.

Now let $\mathit{vetoes}(c) < \ell$. In this case, the briber can make c reach zero vetoes by bribing all $\mathit{vetoes}(c)$ voters vetoing c. Similarly to the previous case (setting $\bar{v} = 0$), each c_j must have at least $n - K + 1$ vetoes in the final election. Otherwise, the lottery can pick K voters and none of them vetoes c_j. These preconsiderations lead to the following greedy algorithm:

- Bribe every voter vetoing c, give the veto to the c_j with the currently smallest veto number. Update the veto number of c_j and repeat this step until all $\mathit{vetoes}(c)$ vetoes for c are replaced by vetoes for non-distinguished candidates.
- Check whether c is a necessary unique winner. If yes, we accept. If no, bribe a voter vetoing a candidate with the currently highest veto number and assign the veto to a candidate with the presently lowest veto number. Do this at most $\ell - \mathit{vetoes}(c)$ times. Check in every step whether c is a necessary unique winner, i.e., whether $\mathit{vetoes}(c_j) \geq n - K + 1$ holds for each j.

[11] The intuition behind this is as follows: For $\mathit{vetoes}(c_j) = n - K + \bar{v}$, the lottery can ignore $n - \bar{v}$ vetoes of c_j and draw only $\bar{v}$ voters vetoing c_j. Hence, c_j and c have the same veto number $\bar{v}$ for some K-voter subelections.

Verify that our problem is surely in P because we merely have to check some inequalities and apply a greedy subroutine in the last subcase. ❑

Next, we show that necessary bribery is easy for 2-Veto as well.

Theorem 4.24. LOTTHEN2-VETO-NECESSARY BRIBERY *is in* P.

Proof. Our algorithm distinguishes the following two cases. In each case, the briber bribes voters vetoing c with highest priority and each bribed voter does not veto c after the bribery.

- $\mathit{vetoes}(c) \geq \ell$. In this case, the briber bribes only voters vetoing c. After the bribery, there are $\bar{v} := \mathit{vetoes}(c) - \ell \geq 0$ voters vetoing c. We check whether c is the winner for every K-voter subelection when all these $\bar{v}$ voters are drawn by the lottery. First of all, if $\bar{v} \geq K$, we reject as there are K-voter subelections where all voters veto c. Hence, we let $0 \leq \bar{v} \leq K-1$ in the following. Next we show that c is a necessary winner after the bribery if and only if each c_j has at least $n-K+\bar{v}+1$ vetoes in the final election. We argue similarly to the arguing in the proof of Lemma 4.18 and use the identity $\bar{s} = K - \bar{v}$, by interchanging the roles of vetoes and approvals. More precisely, c has at most $\bar{v} \in \{0,\dots,K-1\}$ vetoes in any K-voter subelection according to our assumption. Therefore, c has at least $\bar{s} = K - \bar{v} \in \{1,\dots,K\}$ points in any subelection. From Lemma 4.18, we know that each c_j may have at most $\bar{s}-1$ approvals in total. Thus, c_j must have at least $n-(\bar{s}-1)$ vetoes. Exploiting again that $\bar{s} = K - \bar{v}$, we obtain that c_j must have $n-(K-\bar{v}-1) = n-K+\bar{v}+1$ or more vetoes in the final election.

  Thus, we accept if and only if the briber reaches that every c_j has $n-K+\bar{v}+1$ or more vetoes in the final election. To check whether such a bribery exists, we define a greedy algorithm similar to the one in the proof of Theorem 3.26. In contrast to the other proof, each vote is complete in our lottery-based setting and the briber must ensure that each non-distinguished candidate has $n-K+\bar{v}+1$ vetoes in total (instead of $\mathit{pvetoes}(c)-\ell$ vetoes in the proof with partial votes). The remainder of the proof works analogously. The briber bribes ℓ voters vetoing c and each bribed voter initially vetoes c and a candidate c_j with currently the largest veto number among all non-distinguished candidates. After ℓ such bribes, it remains to check whether or not the bribed voters can assign 2ℓ vetoes in total and at most ℓ vetoes to each c_j such that each c_j has $n-K+\bar{v}+1$ or more vetoes in the final election.

- $\mathit{vetoes}(c) < \ell$. Then c can reach zero vetoes, but—if necessary—the briber bribes further voters initially not vetoing c. We thus may fix the bribes concerning the $\mathit{vetoes}(c)$ voters originally vetoing c. The question is which further voters the briber should bribe and whether it makes sense at all to bribe any further voters. A necessary and sufficient condition for a successful bribery is again that—since $\bar{v} = 0$—each c_j has $n-K+1$ or more vetoes after the bribery. If some c_j has fewer vetoes, he still needs a positive number of additional vetoes from the bribed voters. If c_j has exactly $n-K+1$ vetoes (without c being simultaneously vetoed), c_j does not require any new vetoes, but must not lose any further vetoes by bribery. Thus we proceed as follows:

  First bribe all voters vetoing c. Check whether these $2\mathit{vetoes}(c)$ vetoes can be assigned to the c_j such that $\mathit{vetoes}_{(C,\bar{V})}(c_j) \geq n-K+1$ after the bribery for each $j \in [m-1]$. This is equivalent to

$$n-K+1-\mathit{vetoes}(\neg c,c_j) \leq \mathit{vetoes}(c) \quad \forall j \in [m-1]$$

and

$$\sum_{j=1}^{m-1} \max(0,n-K+1-\mathit{vetoes}(\neg c,c_j)) \leq 2\mathit{vetoes}(c).$$

If yes, we accept (cf. Lemma 3.24). Otherwise, we define the following b-edge matching. We are given an undirected multigraph $G=(\mathscr{V},\mathscr{E})$ where $\mathscr{V}$ contains all candidates c_j ($j \in [m-1]$). Each voter vetoing two candidates in $\mathscr{V}$ yields an edge $e \in \mathscr{E}$ connecting these two candidates. Finally we have upper constraints $b(c_j) := \max(0,\mathit{vetoes}(\neg c,c_j)-(n-K+1))$. The capacity constraints are nonnegative and in particular, they are only positive when $\mathit{vetoes}(\neg c,c_j)-(n-K+1)>0$, that is, c_j may lose a positive number of vetoes and has still enough vetoes to lose against c in all K-voter subelections. In other words, c_j has at least two vetoes in each subelection of K voters and may lose some vetoes, given that $b(c_j)>0$. Observe that any matching may contain only edges where both candidates may still effectively lose further vetoes. We compute a maximum matching with Δ edges. If $\Delta+\mathit{vetoes}(c) \geq \ell$, the briber bribes $\ell-\mathit{vetoes}(c)$ arbitrary voters according to edges in the matching and all $\mathit{vetoes}(c)$ voters initially vetoing c. Note that the briber can effectively replace 2ℓ vetoes. It remains to check whether the briber can assign at most ℓ vetoes to each c_j and at most 2ℓ vetoes in total such that each c_j has a total of at least $n-K+1$ vetoes.

If $\Delta+\mathit{vetoes}(c)<\ell$, the briber bribes all voters according to the edges in a maximum matching. By setting $E := \sum_{j=1}^{m-1} b(c_j)$ (the total excess of vetoes that badly performing candidates may still lose in votes initially not vetoing c), the briber can bribe $\min(E-2\Delta,\ell-\mathit{vetoes}(c)-\Delta)$ voters where exactly one vetoed candidate different from c may lose yet another veto and the other candidate not (the briber additionally bribes all $\mathit{vetoes}(c)$ voters vetoing c); otherwise, the edge would belong to the matching and the cardinality of the maximum matching would be larger than Δ. Note that again it does not matter according to which maximum matching the briber bribes the voters and which further voters he bribes (where only one veto can be effectively replaced) as he leaves the veto for the "strong" candidate unchanged. Otherwise, this one would require a new veto in turn. Note that if $E-2\Delta<\ell-\mathit{vetoes}(c)-\Delta$, the briber could nevertheless bribe further voters to completely exhaust his bribery limit, but he cannot effectively replace these vetoes with vetoes that strong candidates require.

Again, it remains to check whether the briber can greedily assign the missing vetoes in the bribed votes to candidates who need some further vetoes. Once more, we can make use of Lemma 3.24 to do so.

Observe that both cases can be decided in polynomial time as our problem by and large reduces to applying a greedy algorithm, computing a maximum matching, and greedily assigning vetoes to non-distinguished candidates. ❑

We here point out that the previous proof provides—under $K=n$—yet another proof for 2-VETO-BRIBERY (under full information). Compared to the proof in [119] (we applied a greedy algorithm and a matching approach in contrast to Lin) showing the result merely for the co-winner model, our proof tailored to the unique-winner case is much more involved and requires additional

considerations in case of $\mathit{vetoes}(c) < \ell$. In opposition to the co-winner model, it may be necessary that the briber bribes some voters not vetoing c in the unique-winner case.

Our next result holds due to Theorem 4.17 and 3.27 where it is shown that standard bribery is a special case of necessary bribery in lot-based voting rules and bribery in 3-Veto is NP-complete under the unique-winner model, respectively. For the co-winner model, we know from [119] that bribery in k-Veto is hard for $k \geq 4$.

Corollary 4.25. LOTTHENk-VETO-NECESSARY BRIBERY *is* NP*-complete for* $k \geq 3$ *in the unique-winner model and for* $k \geq 4$ *in the co-winner model.*

In contrast, necessary bribery in 3-Veto is easy under the co-winner model.

Theorem 4.26. LOTTHEN3-VETO-NECESSARY BRIBERY *is in* P *in the co-winner model.*

Proof. First, if $\mathit{vetoes}(c) \leq \ell$, the briber can bribe all voters vetoing c. As c has zero vetoes in total after the bribery, the same holds for each K-voter subelection. Consequently, we accept.

For $\mathit{vetoes}(c) > \ell$, the briber reaches that c has $\bar{v} := \mathit{vetoes}(c) - \ell (> 0)$ vetoes in the final election. If $\bar{v} \geq K$, we reject since there are K-voter subelections such that all voters veto c. Due to $m > k$, there are three or more other candidates and one of them necessarily has fewer than K vetoes in a given K-voter subelection. Hence, c is never a winner in such elections.

Henceforth, we let $0 < \bar{v} < K$. Following the arguing in the proof of Theorem 4.24, the briber can make c a necessary winner if and only if each c_j has at least $n - K + \bar{v}$ vetoes in the final election (which is equivalent to the condition that at most $K - \bar{v}$ voters do not veto c_j after the bribery; the proof of Lemma 4.18 can be easily adapted to the co-winner model in a way that each c_j must have at least $n - K + \bar{v}$ vetoes or at most $\bar{s}$ points in total, depending on whether the underlying voting rule is k-Veto or k-Approval).

We regard the following generalized b-edge cover problem (suppose that $V^{\neg c}$ contains all voters not vetoing c, and V^c the voters vetoing c). Each c_j yields a vertex c_j ($1 \leq j \leq m-1$). There is further a vertex b assigning 3ℓ vetoes to non-distinguished candidates in the bribed votes. The edges are defined as follows:

- Each voter in V^c vetoing c, c_i and c_j yields an edge between c_i and c_j ($1 \leq i, j \leq m-1$, $i \neq j$).
- For each $j \in [m-1]$, there are ℓ edges $\{b, c_j\}$.
- The capacities are $b_l(c_j) = \max(0, n - K + \bar{v} - \mathit{vetoes}_{(C,V^{\neg c})}(c_j))$ ($j \in [m-1]$), $b_l(b) = b_u(b) = 3\ell$. All remaining upper capacities are unrestricted.

Analogously to the proof of Theorem 3.29 (bribery in 3-Veto under partial information in the co-winner model), it follows that the briber can make c a necessary winner if and only if there is a cover with $3\ell + \bar{v} = \mathit{vetoes}(c) + 2\ell$ edges. (Note that the cover—provided that one exists—contains 3ℓ edges incident to b and any number between zero and $|V^c|$ edges according to voters in V^c. If only $\bar{v}$ edges—and the vetoes for non-distinguished candidates assigned by the corresponding voters—are in the cover according to voters in V^c left unchanged, there is a successful bribery. Otherwise, we need to account for the vetoes assigned by more than $\bar{v}$ voters in V^c in order to make c a necessary winner and this contradicts to our assumption that exactly ℓ voters are bribed. The capacity constraints ensure that each c_j has at least the required number of vetoes, where we fix all

vetoes from (unchanged) voters in $V^{\neg c}$. Zero constraints are trivially met by each cover and express that c_j even gets enough vetoes from voters in $V^{\neg c}$.

Observe that the only differences between the two proofs lie in the capacity constraints for the c_j and in the fact that there are no edges of type $\{c_j, *\}$ or $\{*, **\}$ as it is the case for necessary bribery with partial votes.

Since calculating a minimum edge cover is easy, our original problem is easy as well. ❑

Note that we obtain yet another instance of ties matter as the corresponding problem is hard in the unique-winner case. And once again the voting rule involved is 3-Veto.

In contrast to necessary bribery, the possible bribery problems asks whether we can find a K-voter subelection for which c is the unique winner after the bribery. Let us first consider the Plurality rule.

Theorem 4.27. LOTTHENPLURALITY-POSSIBLE BRIBERY *is in* P.

Proof. We may assume that the lottery picks as many votes for c as possible. Hence, after the bribery there are K-voter subelections giving c a score of $\min(K, \mathit{score}(c)+\ell)$ points.

First, if $\mathit{score}(c)+\ell > \frac{K}{2}$, there exists a K-voter subelection such that more than half of all voters favor c and—as no other candidate achieves the same score as c in this subelection—c is the only winner and hence a possible winner. Our algorithm therefore outputs YES in this case.

In case $\mathit{score}(c)+\ell \leq \frac{K}{2}$, our distinguished candidate c is only the winner of a subelection of K voters if the lottery can select $K-(\mathit{score}(c)+\ell)$ voters supporting other candidates and none of them reaches a final score greater than $\mathit{score}(c)+\ell-1$ in this subelection. Each c_j can contribute with at most $\min(\mathit{score}(c_j), \mathit{score}(c)+\ell-1)$ points to a subelection where c is supposed to be the sole winner with $\mathit{score}(c)+\ell$ points (this number may be even smaller after the bribery). We refer to these points as *harmless points*, as, being drawn by the lottery, they do not prevent c from being the winner of the subelection. Therefore, a rational briber should not take away points from c_j "below the threshold" $\mathit{score}(c)+\ell-1$ when there is a way to bribe a voter supporting a candidate with at least $\mathit{score}(c)+\ell$ point. This leads to the same greedy algorithm as under full information [79]. The briber step by step bribes a voter voting for a c_j with the currently highest score. Accordingly, he bribes ℓ voters and each bribed voter gives the point to c. After performing this greedy procedure, it remains to check whether $K-\mathit{score}(c)-\ell \leq \sum_{j=1}^{m-1} \min(\mathit{score}(c)+\ell-1, \mathit{score}_{(C,\bar{V})}(c_j))$, where $\mathit{score}_{(C,\bar{V})}(c_j)$ denotes the final score of c_j after the briber has bribed ℓ voters according to the proposed greedy algorithm. The inequality expresses that for our K-voter subelection we can find $K-\mathit{score}(c)-\ell$ voters not supporting c such that no c_j has more than $\mathit{score}(c)+\ell-1$ points in this subelection. ❑

The next proof—for 2-Approval—is a little more involved although the problem is yet in P.

Theorem 4.28. LOTTHEN2-APPROVAL-POSSIBLE BRIBERY *is in* P.

Proof. Our algorithm distinguishes the following cases. First, if $\mathit{score}(c)+\ell \leq 1$ or $K=1$, we reject as in any subelection, c is approved of at most once. Arguing similarly to the proof for necessary bribery (cf. Theorem 4.20), it follows that c is not a possible unique winner then.

Second, in case $\mathit{score}(c)+\ell \geq K > 1$, the briber can ensure a full approval score for some subelections. We accept except for $\ell = 0$ and $\mathit{score}(c) = \mathit{score}(c, c_j) \geq K$ for some c_j (that is, all

voters approving of c approve of the same c_j as well and the briber cannot change this). Otherwise, there are K-voter subelections where c is approved by K voters and at least two different candidates are approved aside from c.

It remains to argue for the case where $2 \leq score(c) + \ell < K$. We may assume in this case that the lottery picks all voters approving of c after the bribery—as this is the best case for c—and there are at least two and at most $K-1$ such voters. Again, we reject in case $score(c) = score(c, c_j)$ for some c_j and $\ell = 0$ as then the briber cannot make c reach a higher score than c_j for any K-voter subelection. If it holds $score(c) > score(c, c_j)$ for all c_j or $\ell > 0$, we try to find $K - score(c) - \ell$ voters each of whom approves of two non-distinguished candidates so that every c_j has at most $score(c) + \ell - 1$ points in this K-voter subelection. As candidate c_j shares $score(c, c_j)$ common approvals with c, at most $score(c) + \ell - score(c, c_j) - 1$ unbribed voters disapproving of c and approving of c_j may join this subelection. This leads to the following b-edge matching problem:

Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected multigraph with vertices $\mathcal{V} = C \setminus \{c\}$ and edges as follows. Each voter approving of c_i and c_j ($1 \leq i, j \leq m-1$, $i \neq j$) yields an edge $\{c_i, c_j\}$ in $\mathcal{E}$. Moreover, vertex c_j has upper capacity $b(c_j) = score(c) + \ell - score(c, c_j) - 1$.

Arguing similarly to the proof of Theorem 4.20, it follows that the briber can make c a winner for at least one K-voter subelection if and only if there is an edge matching with $K - score(c) - \ell$ edges and the following inequality holds:

$$\left(\sum_{j=1}^{m-1} (score(c) + \ell - score(c, c_j) - 1)\right) - 2(K - score(c) - \ell) - \ell \geq 0.$$

There are only two slight differences:

1. The capacity constraints reflect that c has a score of $score(c) + \ell$ in this subelection (compared to the proof of Theorem 4.20 according to which c has only $\bar{s}$ points in some worst-case subelections). In other words, the lottery draws as many c-voters as possible (instead of as few c-voters as possible).

2. In contrast to the previous case, where all $K - \bar{s} = n - score(c) - \ell$ voters disapproving of c after the bribery are picked by the lottery, now only $K - score(c) - \ell$ of these voters (and their approvals for non-distinguished candidates) are considered in a K-voter subelection giving c the highest potential score.

Again, each voter approving of two non-distinguished candidates one-to-one corresponds to an edge in the graph. A matching with this cardinality must necessarily exist as otherwise the lottery cannot fill this subelection with exactly $K - score(c) - \ell$ voters not approving of c and c is the sole winner with $score(c) + \ell$ points (as some c_j's capacity constraint is hurt and hence c_j has no less points than c in the given K-voter subelection, even without accounting for the points for c_j from bribed voters approving of c_j and c after the bribery). Likewise, the inequality expresses that the briber can assign ℓ approvals to non-distinguished candidates and c is still the only winner for at least one K-voter subelection. ❑

From Theorem 4.17 and since bribery in 3-Approval is hard under full information [119] (i.e., when the lottery selects all voters), it follows that LOTTHENk-APPROVAL-POSSIBLE BRIBERY is NP-complete for every $k \geq 3$ (see also Theorem 4.21 for the adaption to the unique-winner model):

Corollary 4.29. LOTTHENk-APPROVAL-POSSIBLE BRIBERY *is* NP*-complete for* $k \geq 3$.

Let us turn to the Veto rule.

Theorem 4.30. LOTTHENVETO-POSSIBLE BRIBERY *is in* P.

Proof. First, assume that $K < m-1$. We reject because even though no voter vetoing c is drawn by the lottery, there is at least one non-distinguished candidate not being vetoed either. Hence, c cannot be the only winner in any subelection with K voters.

Henceforth, we let $K \geq m-1$. If $\mathit{vetoes}(c) - \ell \leq 0$, the briber can make c reach a final veto number of zero. However, in contrast to the co-winner model, this does not imply the victory for c as there may be too many candidates who end up in zero vetoes, too. Let $C_0 := \{c_j : \ 1 \leq j \leq m-1, \ \mathit{vetoes}(c_j) = 0\}$ and $c_0 := |C_0|$. We claim that our instance is a YES instance if and only if $\ell \geq c_0$ holds.

($\Rightarrow$): If there exists a possible bribery, the briber can ensure that each c_j has more than zero vetoes after the bribery. This implies that the bribery limit is so large that each c_j with initially zero vetoes can be assigned one veto.

($\Leftarrow$): Suppose that $\ell \geq c_0$. Then a successful bribery strategy is given as follows:

1. Bribe all voters vetoing c and make as many as possible among them veto pairwise different candidates in C_0.

2. Now two cases have to be distinguished from each other:

   (a) If $\mathit{vetoes}(c) \leq c_0 (\leq \ell)$, bribe further $c_0 - \mathit{vetoes}(c)$ voters initially vetoing the currently worst c_j. We argue next that, according to this greedy algorithm, the briber never bribes voters vetoing a non-distinguished candidate with exactly one veto, i.e., there are at least $c_0 - \mathit{vetoes}(c)$ vetoes "above" one for non-distinguished candidates in the original election. Or, equivalently speaking, it holds $c_0 - \mathit{vetoes}(c) \leq \sum_{j=1}^{m-1} \max(0, \mathit{vetoes}(c_j) - 1)$. Note that c initially has $\mathit{vetoes}(c)$ vetoes, c_0 other candidates have zero vetoes, and thus $m-1-c_0$ non-distinguished candidates are vetoed by $n - \mathit{vetoes}(c)$ voters in total. Due to our assumptions, each of these candidates has at least one veto; otherwise, they would belong to C_0. This implies that $n - \mathit{vetoes}(c) \geq (m-1-c_0) \cdot 1$. It remains to show that even $n - \mathit{vetoes}(c) \geq (m-1-c_0) + (c_0 - \mathit{vetoes}(c))$ holds. This latter inequality expresses that each of the $m-1-c_0$ candidates in $C \setminus (\{c\} \cup C_0)$ has at least one veto and their total number of vetoes "above one" is at least $(c_0 - \mathit{vetoes}(c))$ (i.e., the number of voters not vetoing c who are bribed). This inequality is equivalent to $n \geq m-1$ which is true because of the above assumption $n \geq K \geq m-1$.
   All in all, the briber greedily bribes $c_0 - \mathit{vetoes}(c)$ voters vetoing the currently worst non-distinguished candidate (with presently at least two vetoes) and each bribed voter vetoes a candidate in C_0 not yet vetoed in any vote bribed up to now.

   (b) If $\mathit{vetoes}(c) > c_0$, the briber bribes all voters vetoing c. c_0 of them veto the same number of candidates with initially zero vetoes. The remaining bribed voters assign vetoes to arbitrary non-distinguished candidates. Now each c_j has at least one veto in the final election. Since $K \geq m-1$, the lottery can pick one veto for each c_j and arbitrary $K - m + 1$ further vetoes for candidates other than c. It follows that c is the only winner (with zero vetoes) in this K-voter subelection.

Next regard the case where $vetoes(c) - \ell > 0$ and $vetoes(c) - \ell \leq n - K$ hold. Now c can reach zero vetoes for some K-voter subelection after the bribery. In contrast to the previous case, the briber bribes only voters vetoing c and some voters vetoing c remain unchanged (we assume that these voters are not selected by the lottery). Once more, a reasonable briber step by step assigns the current veto to the presently best c_j. Accordingly, he reaches that as many candidates as possible with zero vetoes in the initial election have a veto in the final election. Again, we accept if and only if $\ell \geq c_0$ holds. Remind that, due to our above settings, it holds $K \geq m-1$, that is, K is large enough that the lottery can select at least one veto for each non-distinguished candidate.

Last but not least, for $vetoes(c) - \ell > n - K$, c has at least $\bar{v} := vetoes(c) - \ell - n + K > 0$ vetoes. Again, the briber does best by greedily assigning each veto to the currently best non-distinguished candidate. In the end, all non-distinguished candidates' vetoes and exactly $\bar{v}$ vetoes for c are picked by the lottery and it remains to verify whether or not c is the only winner. As a combined result, we therefore accept if and only if $K \geq \bar{v} + (m-1)(\bar{v}+1)$ and $\sum_{j=1}^{m-1} \max(0, \bar{v}+1-vetoes(c_j)) \leq \ell$. The first inequality reflects that the lot size K is great enough to ensure that at the same time, c may have $\bar{v}$ vetoes and all other candidates (at least) $\bar{v}+1$ vetoes at all. The second condition states that the bribery limit is large enough that bribed voters can assign to each c_j (at least) $\max(0, \bar{v}+1-vetoes(c_j))$ additional vetoes such that c_j has more vetoes than c in this K-voter subelection.

Observe that all cases are in P as they reduce to checking some (polynomially restricted) inequalities or applying some (polynomially bounded) greedy subroutines. ❑

Next we find that our problem is easy for 2-Veto, too.

Theorem 4.31. LOTTHEN2-VETO-POSSIBLE BRIBERY *is in* P.

Proof. Our algorithm distinguishes the following cases:

- If $\ell \geq K$, the bribery limit is so large that the lottery can draw K-voter subelections merely consisting of bribed voters. We accept if and only if $2K \geq m-1$ since then and only then the lot size is large enough to ensure that all c_j are vetoed once.

- If $K > \ell$ and $vetoes(c) - \ell \leq n - K$, the briber bribes ℓ voters such that as many voters vetoing c as possible are bribed, there are K-voter subelections after the bribery such that no voter vetoes c, and the briber cannot determine a complete subelection with ℓ bribes. W.l.o.g., we assume that all bribed voters are picked by the lottery. We regard a generalized b-edge cover problem defined as follows. Let $G = (\mathscr{V}, \mathscr{E})$ be an undirected multigraph with vertices $\mathscr{V} = (C \setminus \{c\}) \dot{\cup} \{b\}$. The edges in $\mathscr{E}$ are defined as follows:

  - Each voter vetoing c_i and c_j ($i, j \in [m-1], i \neq j$) yields an edge $\{c_i, c_j\}$.
  - For each $j \in [m-1]$, there are ℓ edges $\{b, c_j\}$.

  The lower capacities are $b_l(c_j) = 1$ ($1 \leq j \leq m-1$), and $b_l(b) = 2\ell$. Moreover, we have $b_u(b) = 2\ell$. All other upper capacities are unrestricted.

  It follows that the briber can make c the winner of at least one K-voter subelection if and only if there is a cover with $\ell + K$ edges. Observe that the basic ideas and the argument of correctness are similar to the ones in the proof of Theorem 4.26. The differences are that each

c_j requires one veto, bribed voters assign 2ℓ vetoes to non-distinguished candidates (instead of 3ℓ vetoes), and we try to find a cover with $K+\ell$ edges, according to K voters. Note that any cover, provided that one exists, contains exactly 2ℓ edges incident to b (corresponding to ℓ bribed voters and the 2ℓ vetoes assigned by them after the bribery) and $K-\ell$ edges $\{c_i, c_j\}$ according to $K-\ell$ voters vetoing two non-distinguished candidates and being left unchanged by the briber. In contrast, in the proof of Theorem 4.26 each voter worth bribing vetoes c and two non-distinguished candidates before the bribery. All in all, the briber bribes arbitrary $\min(\mathit{vetoes}(c), \ell)$ voters initially vetoing c and arbitrary $\ell - \min(\mathit{vetoes}(c), \ell)$ voters not vetoing c and not corresponding to any edge $\{c_i, c_j\}$ in the matching (where $1 \leq i, j \leq m-1$, $i \neq j$).

- $\mathit{vetoes}(c) - \ell > n - K$ ($\Rightarrow \ell < K$; otherwise, if $\ell \geq K$, the inequality $\mathit{vetoes}(c) - \ell > n - K$ is equivalent to $\mathit{vetoes}(c) - n > \ell - K$, and the nonpositive left-hand side is larger than the nonnegative right-hand side which is a contradiction). In this case, the briber cannot reach zero vetoes for c in any subelection. Hence, in all subelections with K voters, c has at least $\bar{v} := \mathit{vetoes}(c) - \ell - n + K (> 0)$ vetoes. If $\bar{v} = K$, we immediately reject. Given that $\bar{v} < K$, we argue as follows. Suppose that c has exactly $\bar{v}$ vetoes in a K-voter subelection. We construct a subelection as follows:

  1. Fix all $n - \mathit{vetoes}(c)$ voters who initially do not veto c. If c is supposed to win in the subelection, these voters should belong to it.
  2. Add greedily $\bar{v}$ voters to the subelection who give a veto to c and for whom the second veto candidate is as good as possible concerning the veto numbers after step 1 (i.e., this non-distinguished candidate currently has as few vetoes as possible). This strategy leads to as uniform veto numbers as possible.
  3. Now ℓ arbitrary voters with c initially being vetoed are bribed and added. If possible, the briber step by step gives the required numbers of missing vetoes to all c_j (at most ℓ additional vetoes per candidate) such that c_j has at least $\bar{v}+1$ vetoes in the constructed K-voter subelection.

  Verify that this subelection consists of exactly K voters as it holds $(n - \mathit{vetoes}(c)) + \bar{v} + \ell = n - \mathit{vetoes}(c) + (\mathit{vetoes}(c) - \ell - n + K) + \ell = K$ according to the three listed voters groups. Either this latter subroutine leads to c being the unique winner with $\bar{v}$ vetoes or not, but if one K-voter subelection with unique winner c exists, this subroutine finds it.

Observe that all three subcases are in P since we check a (polynomially bounded) inequality, compute a minimum edge cover, and apply a greedy procedure, respectively. ❑

Our next result holds due to Theorem 4.17 and because standard bribery (with $K = n$ in our lot-based settings) is NP-complete for k-Veto ($k \geq 3$ in the unique-winner and $k \geq 4$ in the co-winner model). These hardness results under complete information follow from Theorem 3.27 (unique-winner) and from [119] (co-winner).

Corollary 4.32. LOTTHENk-VETO-POSSIBLE BRIBERY *is* NP*-complete for* $k \geq 3$ *in the unique-winner model and for* $k \geq 4$ *in the co-winner model.*

Now just the complexity of LOTTHEN3-VETO-POSSIBLE BRIBERY in the co-winner model has not been considered yet. Our final theorem presents yet another case where ties matter.

Theorem 4.33. LOTTHEN3-VETO-POSSIBLE BRIBERY *is in* P *in the co-winner model.*

Proof. First of all, if $\mathit{vetoes}(c) - \ell \leq n - K$, we accept since the briber can ensure that there are at least K voters who do not veto c which in turn implies that c is a winner with zero vetoes for at least one K-voter subelection.

Hence, it remains to consider the case where $\mathit{vetoes}(c) - \ell > n - K$. In this case, c receives $\bar{v} := \mathit{vetoes}(c) - \ell - n + K > 0$ vetoes for sure. Let $\bar{v} < K$ because otherwise c cannot be a winner for any subelection of K voters. We assume that the lottery selects all voters initially not vetoing c (we denote these voters by $V^{\neg c}$). There are $K - \bar{v} (= n - \mathit{vetoes}(c) + \ell)$ such voters in total after the bribery. Note that $n - \mathit{vetoes}(c)$ voters do not veto c before and after the bribery, while ℓ voters veto c before but not after the bribery. Consequently, each c_j has thus at least $\mathit{vetoes}_{(C,V^{\neg c})}(c_j)$ vetoes in such subelections and we must find at least $\max(0, \bar{v} - \mathit{vetoes}_{(C,V^{\neg c})}(c_j))$ further voters belonging to our K-voter subelection who (1) veto c and c_j before and after the bribery or (2) are bribed and veto c_j (and surely not c) after the bribery. We regard a generalized b-edge cover problem defined as follows. We are given an undirected multigraph $G = (\mathscr{V}, \mathscr{E})$ with vertices $\mathscr{V} = (C \setminus \{c\}) \dot{\cup} \{b\}$. Moreover, our graph contains the following edges $\mathscr{E}$:

- Each voter vetoing c, c_i, and c_j yields an edge $\{c_i, c_j\}$ in $\mathscr{E}$.
- For each $j \in [m-1]$, there are ℓ edges $\{b, c_j\}$ in $\mathscr{E}$.

The capacities are $b_l(c_j) = \max(0, \bar{v} - \mathit{vetoes}_{(C,V^{\neg c})}(c_j))$ (note that such capacities can be zero, but then candidate vertex c_j trivially meets its lower capacity constraint; in this case, c_j is vetoed by $\bar{v}$ or more voters in $V^{\neg c}$ and does not beat c in such subelections assigning c exactly $\bar{v}$ vetoes) and $b_l(b) = b_u(b) = 3\ell$. All remaining upper capacities are unlimited.

Analogously to the proof of Theorem 4.26 showing that LOTTHEN3-VETO-NECESSARY BRIBERY is in P, it follows that the briber can make c a possible winner via bribing ℓ voters (initially vetoing c) if and only if there is an edge cover with $\bar{v} + 3\ell = \mathit{vetoes}(c) + 2\ell - n + K$ edges. One slight difference compared to the other proof is that, in this context, only $\mathit{vetoes}(c) - \ell - n + K$ vetoes for c count in our subelection (instead of $\mathit{vetoes}(c) - \ell$ as in the other proof where as many c-vetoes as possible are considered by the lottery) and therefore all other candidates require only $\mathit{vetoes}(c) - \ell - n + K$ vetoes in total, instead of $\mathit{vetoes}(c) - \ell$. Notice that for possible bribery the lottery selects all voters not vetoing c and as few voters as possible vetoing c.

In a cover, 3ℓ edges correspond to 3ℓ vetoes assigned by ℓ bribed voters to non-distinguished candidates. Moreover, $\bar{v}$ edges one-to-one correspond to the same number of voters vetoing c who are not bribed and who are picked by the lottery.

As computing a minimum edge cover is easy (and the transformation to this problem is in FP), our overall problem turns out to be in P. ❑

Observe that the proofs of Theorem 4.26 and Theorem 4.33, showing that LOTTHEN3VETO-NECESSARY BRIBERY and LOTTHEN3VETO-POSSIBLE BRIBERY are easy under the co-winner model, are generalizations of the proof of 3-VETO-BRIBERY (i.e., under full information) and use basically the same construction as the original proof by Lin [119] (with generalized edge cover instead of classical edge cover). Notice that for $K = n$ both proofs more or less coincide.

4.4 Conclusion

We have presented a detailed complexity analysis for lot-based voting with k-Approval and k-Veto as underlying voting rules. The decision problems of evaluation, of possible winners, and necessary winners have been analyzed. We have also considered the counting complexity and two bribery problems—possible and necessary bribery.

All our results hold under the unique-winner model and the assumption that each combination of K voters is drawn by the lottery with the same probability, most results are still valid when we drop the uniform distribution assumption and allow that each K-voter subelection is drawn by a general positive probability.

While necessary winner is easy even for all scoring rules, possible winner is hard for almost all voting rules belonging to the k-Approval/-Veto families. More precisely, possible winner is easy for $k \leq 2$ and hard for $k > 2$ both for k-Approval and k-Veto in the unique-winner model. As for the counting versions of possible winner, only Plurality and Veto are easy to count, all other problems are #P-complete. As a byproduct of this endeavor, we have identified interesting cases where it is easy to decide but hard to count as well as problems where ties matter, i.e., the complexity differs depending on whether the unique- or co-winner model is used. Moreover, we have provided an FP recursion for VETO-#CCAV. For necessary and possible bribery, we have found that, despite the presence of a lot-based voting rule, the computational (worst-case) complexity does not change. These findings indicate that further voting rules have to be explored and it might become necessary to search for a refined lot-based formalism to impede bribery.

Although there is no additional hardness for bribery under the use of this lot-based voting mechanism, lot-based voting makes it harder for a briber to certainly reach his goal. We provide some examples illustrating that the briber's (or, more precisely, c's) success strongly depends on the lot size K and even though the briber reaches c to be a winner of the overall election, a small lot size—compared to the total number of voters—may make things more difficult to the briber. Consider the following example:

Example 4.34. *Let (C,V) be an election with candidate set $C = \{c, c_1, c_2\}$, voter set $V = \{v_1, \ldots, v_8\}$ (i.e., $m = 3$ and $n = 8$), designated candidate c, and $\mathscr{F} =$ Veto. We have vetoes$(c) = 2$, vetoes$(c_1) =$ vetoes$(c_2) = 3$.*

Table 4.2 presents the probabilities of c being the LOTTHENVETO winner for all possible lot sizes K. We denote the probability of c being the only winner in a K-voter subelection with Prob(K).

K	1	2	3	4	5	6	7	8
Prob(K)	0	$\frac{9}{28}$	$\frac{9}{28}$	$\frac{3}{14}$	$\frac{3}{7}$	$\frac{13}{28}$	$\frac{1}{4}$	1
(in %)	0.00	32.14	32.14	21.43	42.86	46.43	25.00	100.00

Table 4.2: Overview of the winning probabilities of c in Example 4.34 for all lot sizes K.

In the Appendix, we will provide the detailed calculations. Although c is the unique winner of the entire election (that is, for $K = n = 8$), the winning probability of c collapses down to 25 percent when the lottery selects all but one voter. Interestingly, the probability is considerably higher for $K = 6$ (nearly twice the probability for $K = 7$) and again falls a little when resetting $K = 5$. Interestingly, $K = 4$ yields a local minimum of the winning probability (21.42%). Notably,

for $K = 2$ and $K = 3$ the winning probability is identical and about one third, while it falls down to zero for $K = 1$.

We can generalize Example 4.34 in a way that there are two voters vetoing c and for each c_j ($1 \leq j \leq r$) there are three voters vetoing c_j. By this we obtain a sequence (C^r, V^r) of elections (where $|V^r| = 3r+2$ and $|C^r| = r+1$, that is, there are r non-distinguished candidates). For $K = n$, the winning probability is one, while for $K = n-1$ the winning probability is $\frac{2}{3r+2}$. Observe that for $r \to \infty$ the winning probability collapses from 100% down to nearly zero when we select all but one voter instead of evaluating the entire election.

At first sight, we could observe a strange and nearly paradoxical behavior of c's winning probability. One interesting point is that the winning probability collapses from 1 to around $1/4$ (or to $2/(3r+2)$, in the generalization of Example 4.34) when the lottery ignores only one voter. On the other hand, the winning probability does not gradually fall or rise, but reveals a zigzagging behavior. The reason for the first (and possibly for the second aspect as well) is that c is only just a unique winner in the overall election and the election is somewhat unstable. An intriguing task for future research is therefore studying similar examples and finding instances where small, large, or medium lot sizes are beneficial for c, or examining the question whether there are some paradoxes (e.g., that the winning probability is zero if and only if $K = \frac{n}{2}$ or if and only if $K = n$).

In our example, the best possible lot size for c is the largest possible lot size. In other examples, a candidate might be better off when the lottery selects only few voters. Suppose for instance that one voter favors c over all other candidates and 100 voters favor all other candidates over c. Surely, c is not the Condorcet winner in the overall election. In fact, c is a Condorcet loser for each lot size $K > 2$ in every K-voter subelection. Nevertheless, c is a possible Condorcet winner, but only for $K = 1$ when the subelection draws the first voter. All other lot sizes prevent c from being the Condorcet winner. The following example provides an instance where K has to take a "middle" value.

Example 4.35. *Let (C,V) be an election with candidate set $C = \{c, c_1, \ldots, c_7\}$, voter set $V = \{v_1, \ldots, v_9\}$, and distinguished candidate c. Let $\mathscr{F}$=2-Approval. The voters in V are defined as follows:*

1. *v_i approves of c and c_i ($1 \leq i \leq 3$).*
2. *$v_4, \ldots, v_6$ approve of c_4 and c_5.*
3. *$v_7, \ldots, v_9$ approve of c_6 and c_7.*

In the Appendix, we will show that c is a possible winner if and only if $2 \leq K \leq 7$.

Examples 4.34 and 4.35 raise the question which lot size is best possible for an external agent—a briber or a chair—to reach his goal. Although an agent cannot make c a necessary winner with his given budget, he might be able to maximize the winning probability of c whenever the agent has influence on the lot size and/or some voters. Going back to the question whether lotteries offer an additional protection against strategic behavior, especially Example 4.34 supports this conjecture, though not in terms of worst-case complexity: as soon as some voters do not belong to this subelection, the winning probability of c for several exemplary values $K < n$ is smaller than 50%. Hence, even though the briber can make c (only just) a unique winner with his resources, as in Example 4.34

(where we may suppose that this is the final election after some voters have been bribed), the lottery jeopardizes c's chances of success.

In contrast to our examples, the winning probability is even hard to determine for general K and n when the underlying voting rule is k-Approval or k-Veto for $k \geq 2$. This is a mere consequence of the fact that the corresponding counting problem is hard then. In order to determine the winning probability of c, we need the number of subelections with c as the winner. Additionally, a chair or briber with influence on the lot size faces the problem that sometimes a small lot size is favorable for c, for other instances a lot size "in between" maximizes c's winning probability, while in other examples the lot size should be as large as possible. Coming back to the question whether lot-based voting excludes or at least impedes strategic behavior in elections, we have to point out that lot-based voting does not make things harder to a briber or a chair in terms of worst-case complexity, but one can say that additional difficulties arise for a manipulative agent, especially when the counting problem is hard in general (i.e., for k-Approval/-Veto and $k > 1$). We refer to studying lot size control models combined with bribery or multimode control attacks for future research. Investigating the winning probability (or winning probability distribution over all candidates in C) therefore appears to be a fascinating task, both under the uniform distribution assumption and for general probabilities.

Whether or not the randomization step first running a lottery on the voter set satisfies some crucial fairness properties and is hence a good alternative to "classical" voting, cannot be finally accessed at this point. Intuitively, the additional randomization step may be a great disadvantage to strong candidates and a chance for weak candidates. For instance, consider a Plurality election with two candidates c and d, 98 voters supporting c, and 2 voters favoring d. Yet d can be the unique winner in 3-voter subelections. Other, similar examples can be easily constructed for other voting rules as well. Nevertheless, such paradoxical situations can be better circumvented by election designers simply via using more appropriate and more representative lot sizes. We further point out that lotteries defined as in this chapter treat all voters and candidates equally. For a brief discussion of some fairness properties like anonymity or neutrality, we refer to [160] and leave more detailed investigations for future research.

We further refer to the open cases for LOTTHEN$\mathscr{F}$-EVALUATION (for 2-Approval and 2-Veto) for future research. Besides, regarding all problems under the co-winner model appears to be a promising task.

Two problems worth studying are *Improving Bribery* and *Fixed Bribery*. [12] In improving bribery, we ask whether a briber can raise the winning probability of a distinguished candidate by bribing some voters. Note that improving bribery reduces to the possible bribery problem. Suppose that a candidate c is not a possible winner. Then an improving bribery exists if and only if the possible bribery problem has a solution. Moreover, fixed bribery generalizes the evaluation problem and asks whether a briber can make a candidate the winner with a probability larger than a given threshold p. Fixed bribery generalizes standard bribery, necessary, and possible bribery and hence we can derive many hardness results all at once.

One could also investigate other kinds of strategic behavior such as multimode control. Furthermore, one could consider destructive variants where we check whether the briber can prevent c from being the only winner or among the winners for all, at least one, or at least a certain number of K-voter subelections.

[12] The names remind of the similarly defined problems *Improving Manipulation* and *Fixed Manipulation* [160].

Again we refer to other generalizations such as extending our studies to other voting rules or introducing weights and/or price-tags (when studying bribery). As pointed out above, one could regard a more general model where every combination of K voters is drawn with an individual (positive) probability. Most results still hold (namely, all results in this chapter but the ones for evaluation). For general probabilities, the complexity of evaluation is questionable (e.g., for Plurality our proof exploiting the number of subelections with unique-winner c cannot be tailored to general probabilities; we need a different approach to solve the more general problem). What we can say is that evaluation is hard for general probabilities whenever the corresponding problem is hard under the uniform distribution assumption. In particular, the question arises whether and how the results change when some combinations of K voters are drawn by the lottery with a probability of zero.

In [160], some suggestions for sampling with general probabilities are given. Possibly, this gives rise to some new control variants where a chair can—completely or only to a certain extent—play around with probabilities of K-voter subelections or each voter himself is drawn by the lottery with an individual likelihood.

As another interesting research direction, we propose a variant of lot-based voting where the chair can fix K' voters, while the lottery draws K'' other voters at random (it holds $K = K' + K''$). For $K' = 0$, we deal with lot-based voting in the classical meaning, whereas $K'' = 0$ corresponds to the case where a chair can entirely determine a K-voter subelection and there is no random sampling in this extreme case. Instead of single voters, one could also study models where K voter groups instead of K single voters are selected by a lottery.

We further refer to parameterized control. Natural parameters are $|C|$, $|V|$, ℓ, K, or the difference $|V| - |K|$.

In particular, when K or $|V| - K$ are constants, we obtain interesting results on first sight. For instance, one can easily observe that many problems become easy for constant K. Since we only have to evaluate $\binom{n}{K}$ subelections with K voters, all counting problems, evaluation, possible winner, and necessary winner become/are easy—even for voting rules with an easy winner problem. The same reasoning can be applied to settings with constant $n - K$. All these results still hold when each K-voter subelection is drawn by an individual positive probability.

Moreover, possible bribery in k-Approval and k-Veto becomes easy for all natural numbers k when we let K be constant. Either $\ell \leq K$ holds and we can polynomially enumerate over all $\binom{n}{K}$ K-voter subelections (of which the number is polynomially bounded from above for constant K) and for each such combination, we check whether the briber can make c the winner of the subelection by trying out all $\binom{K}{\ell}$ ways to bribe ℓ out of K voters. If $\ell > K$, ℓ is not constant in general, but the briber can change all votes in some K-voter subelection. We assume then that the briber bribes arbitrary K voters, each of them approves of c, and we simply have to verify whether or not the other approvals can be distributed to non-distinguished candidates in a way that no other candidate has a full approval score. This can be done by a simple greedy algorithm. Note that we can argue analogously for k-Veto.

For constant $n - K$, the complexities for necessary and possible bribery in k-Approval and k-Veto are the same as for the case where $n - K$ and K are unrestricted. This holds as for $K = n$ the difference $n - K$ is constant (namely zero) and hence our problems are at least as hard as bribery under full information.

Once again, we refer to other complexity-theoretic concepts for future research.

Last but not least, lot-based voting is one way to describe uncertainty in voting. One could additionally assume that the votes themselves are partial according to some model in PIM (cf. Chapter 3) and/or the voting rule is not uniquely known (cf. [19]).

Chapter 5

Bribery and Control in Runoff Rules

Up to the present, our focus has been on bribery and winner determination in voting under incomplete information. Nevertheless, there are still many open problems related to voting under full information. In [73], we regarded control and bribery for some voting rules that belong to the most important and frequently studied rules in literature: k-Veto, Copeland$^{\alpha}$, Maximin, Plurality with Runoff, and Veto with Runoff.

As we mainly consider the two families k-Approval and k-Veto in this thesis, we restrict ourselves to the probably most important and novel results in [73] and only focus on Plurality with Runoff and Veto with Runoff. Surprisingly, for both rules neither bribery nor control by adding, deleting, or replacing voters and/or candidates have been investigated in literature so far. That's why we closed the gaps for these voting rules. Once again, we address merely the constructive variants for these problems.

Although we could have also studied k-Approval or k-Veto ($k \geq 2$) in a runoff version, our analysis is limited to the two possibly most prominent rules among them—Plurality with Runoff and Veto with Runoff. Both voting rules work similarly. First of all, we determine the scores and veto numbers according to the Plurality and Veto rule, respectively. The two best candidates then move to a runoff stage and compete against each other. The candidate in the runoff winning the head-to-head contest is the runoff winner. In practice, tie-breaking rules are applied to determine the two runoff candidates and the unique runoff winner. Although we could allow the voters to revote in the runoff, we follow the largest part of the COMSOC literature and assume that all voters truncate their rankings to the two remaining candidates. E.g., when a and b move to the runoff and a voter votes $c \succ a \succ d \succ b \succ e$, the voter must not change his ranking into $b \succ a$ and therefore votes $a \succ b$ in the runoff, too.

The rule Plurality with Runoff and its variations are often used in political elections, e.g., in the presidential election in France. The other rule, Veto with Runoff, occurs—albeit frequently in a multistage version—in competitions of whatever kind with several rounds where in each round the worst candidate is ruled out, e.g., by a jury. Basically, the Veto with Runoff rule and some variations are applied in situations where a committee or an electorate concentrates on the two least unpopular alternatives and ignores all other ones (with more vetoes).

5.1 Related Work

Once more, we refer to Section 3.1 for the literature about bribery and control in elections as well as about the computational aspects of voting rules. In this chapter, we focus on bribery and on multimode control by adding and deleting candidates and voters. Additionally, we address several special cases such as control by adding voters, deleting voters, and replacing voters and study the analogous problems for candidates. We emphasize that the general (multimode) control problem, allowing an external agent to perform different types of control actions at once, such as deleting and/or adding voters and/or candidates, was introduced in [81]. Replacement control was studied in [17] in the context of judgment aggregation; Loreggia et al. [123] considered replacement control in elections. While Faliszewski et al. [81] addressed the voting rules Condorcet, Maximin, Approval, and Plurality, Loreggia et al. mainly focused on replacement control in Borda, Veto, and k-Approval.

As mentioned above, bribery and control have not yet been investigated for the two rules Plurality with Runoff and Veto with Runoff even though both rules apply to many real-world settings. In contrast, the results for coalitional and single manipulation are known to be easy for Plurality with Runoff [170]. Moreover, in parallel to us, Maushagen et al. [125] settled the complexity of shift bribery for several multi-stage voting rules, and in particular for Plurality with Runoff. However, the setting of shift bribery is different from standard bribery and both problems are independent from each other.

Apart from strategic behavior, the Plurality with Runoff rule and/or other runoff and multistage rules and their axiomatic properties were studied in [100, 131, 148] in various directions. The list is far from being exhaustive.

5.2 Results

In this section, we study the complexity of voter control, bribery, and candidate control for Plurality with Runoff and Veto with Runoff. We begin our analysis showing that the problems constructive control by adding voters, by deleting voters, and by replacing voters are easy for both voting rules. After that, we study bribery ere we finally focus on candidate control.

5.2.1 Voter Control

Instead of showing the results separately one-by-one, we prove that *exact constructive control by adding and deleting voters* (denoted by $\mathscr{F}$-ECCAV+DV) is polynomial-time solvable, where $\mathscr{F}$ is either Plurality with Runoff or Veto with Runoff. In this exact variant, our input is an election $(C, V \cup W)$ with $m \geq 2$ candidates in candidate set C, n_V voters in registered voter set V, n_W voters in unregistered voter set W ($n_V, n_W \in \mathbb{N}_0$), a designated candidate c, and two nonnegative integers $\ell_{AV} \leq |W|$ and $\ell_{DV} \leq |V|$ denoting the numbers of voters that may be added from W and deleted from V, respectively. Since we study exact control, we even require that the numbers of added and deleted voters are exactly equal to the corresponding given integers, i.e., we require that $|V'| = \ell_{DV}$ and $|W'| = \ell_{AV}$. Moreover, we have $\ell_{AC} = \ell_{DC} = 0$ and $D = \emptyset$ as we do not consider candidate control in this section. Note that CCAV, CCDV, CCRV, their exact variants, and CCAV+DV are polynomial-time many-one reducible to ECCAV+DV (cf. Theorem 5.3).

If not mentioned other, we assume that ties are broken in favor of candidate c. This way to break ties is sometimes called *parallel-universe tie-breaking* in literature as—given distinguished candidate c—we search for at least one way to break ties over all tied candidates such that our instance is a YES instance for c. In contrast to other tie-breaking schemes, parallel-universe tie-breaking keeps the voting rule anonymous and neutral (as each voter is treated equally and each non-distinguished candidate can win in his own best-case).

Theorem 5.1. PLURALITY WITH RUNOFF-ECCAV+DV *is in* P.

Proof. First regard the case $C = \{c,d\}$ with $d \neq c$. A rational chair adds exactly ℓ_{AV} voters from W, among them as many as possible favor c. Moreover, he deletes precisely ℓ_{DV} voters from V and as many as possible of these voters favor d over c. It remains to check whether c wins or ties with d in the final election or not. Let $n' := |\bar{V}| = n_V + \ell_{AV} - \ell_{DV}$ denote the number of voters in the final election $(C,\bar{V})$. According to the chair's adding/deleting strategy, we accept if and only if

$$score_{(C,\bar{V})}(c) = score_{(C,V)}(c) + \min(\ell_{AV}, score_{(C,W)}(c)) - \max(0, \ell_{DV} - score_{(C,V)}(d)) \geq \frac{n'}{2}$$

as then and only then c is a weak Condorcet winner in the final election $(C,\bar{V})$. Observe that the chair adds exactly $\min(\ell_{AV}, score_{(C,W)}(c))$ voters favoring c from W and deletes precisely $\max(0, \ell_{DV} - score_{(C,V)}(d))$ voters supporting c from V. Hence, $score_{(C,\bar{V})}(c)$ voters prefer c to d in the final election and our algorithm checks whether this number is at least $\frac{n'}{2}$ (otherwise, d wins the pairwise contest against c).

Henceforth, we let $|C| > 2$, that is, not all candidates move to the runoff stage. Our algorithm guesses a candidate $d \in C \setminus \{c\}$ and four integers $\ell^c_{AV}, \ell^d_{AV}, \ell^c_{DV}, \ell^d_{DV}$ such that $0 \leq \ell^c_X + \ell^d_X \leq \ell_X$ for $X \in \{\text{AV}, \text{DV}\}$. The guessed candidate d is supposed to be the one who competes with c in the runoff stage. Moreover, ℓ^c_{AV} (ℓ^d_{AV}) is said to be the number of voters added from W that approve c (d) and ℓ^c_{DV} (ℓ^d_{DV}) denotes the number of voters deleted from V that approve c (d). Given such guessed candidate and integers, we determine whether the chair can add exactly ℓ_{AV} votes of whom ℓ^c_{AV} (ℓ^d_{AV}) approve c (d) and whether he can delete exactly ℓ_{DV} votes of whom ℓ^c_{DV} (ℓ^d_{DV}) approve c (d) such that c and d reach the runoff for at least one way to break ties and c is a runoff winner. Observe that the original instance is a YES-instance if and only if at least one guess leads to a YES answer to the above question. We show how to find the answer to the above question in polynomial-time. We immediately discard the guess if one of the following conditions holds:

1. $\ell^c_{DV} > score_{(C,V)}(c)$,
2. $\ell^d_{DV} > score_{(C,V)}(d)$,
3. $\ell^c_{AV} > score_{(C,W)}(c)$,
4. $\ell^d_{AV} > score_{(C,W)}(d)$.

Assume that none of the above conditions holds. Then, the scores of c and d are fixed. In particular, the final score of $e \in \{c,d\}$ is given by $score_{(C,\bar{V})}(e) = score_{(C,V)}(e) + \ell^e_{AV} - \ell^e_{DV}$. Let $\bar{s} = \min(score_{(C,\bar{V})}(c), score_{(C,\bar{V})}(d))$. To ensure c and d to be in the runoff, each candidate $a \in A :=$

$C \setminus \{c,d\}$ may have at most $\bar{s}$ points in total. A second condition for c to be a runoff winner against d is that c does not lose against d in the pairwise comparison between them. Since there are $n' = n_V + \ell_{AV} - \ell_{DV}$ voters in the final election $(C,\bar{V})$, c must win at least $\lceil \frac{n'}{2} \rceil$ duels against d or, equivalently speaking, at most $\lfloor \frac{n'}{2} \rfloor$ comparisons may be won by d. Let $score_{(C,V)}(A) = \sum_{a \in A} score_{(C,V)}(a)$. Moreover, for $X \in \{AV, DV\}$, let $\ell_X^A = \ell_X - \ell_X^c - \ell_X^d$. (Note that we immediately discard our current guess if $\ell_{AV}^A > score_{(C,W)}(A)$ or $\ell_{DV}^A > score_{(C,V)}(A)$ holds; the conditions $\ell_X^A \geq 0$ ($X \in \{AV, DV\}$) are met due to the definition of ℓ_X^c and ℓ_X^d.) As d in turn wins $score_{(C,\bar{V})}(d)$ comparisons against c in all votes of the final election where d is the top candidate, there may be at most $\lfloor \frac{n'}{2} \rfloor - score_{(C,\bar{V})}(d)$ voters favoring some $a \in A$ and preferring d to c in the final election. Hence, if this number is negative, we immediately reject for the current guess and regard the next one as d beats c in the runoff stage, no matter which voters favoring some $a \in A$ join them. Otherwise, we search for exactly $n_V - score_{(C,V)}(c) - score_{(C,V)}(d) - \ell_{DV}^A = score_{(C,V)}(A) - \ell_{DV}^A$ voters in V not deleted and voting for candidates in A and for exactly ℓ_{AV}^A voters added from W and preferring some $a \in A$ such that the final election contains at most $\lfloor \frac{n'}{2} \rfloor - score_{(C,\bar{V})}(d)$ voters who rank some $a \in A$ first and prefer d over c. This leads to the following mincost flow problem. (For practical reasons, let V^A (W^A) denote the set of voters in V (W) with favorite candidate in A.)

There is a source x, a sink y, and two vertices v^A and w^A. Moreover, each voter in $V^A \cup W^A$ yields a vertex. Likewise, each $a \in A$ yields a vertex a. If not mentioned other, each edge cost is equal to zero. There is an edge from x to v^A with capacity $score_{(C,V)}(A) - \ell_{DV}^A$. There is another edge from x to w^A with capacity ℓ_{AV}^A. Each voter $v \in V^A$ yields an edge (v^A, v) with capacity 1. The cost of this edge is equal to one if and only if v prefers d to c. Analogously we define edges from w^A to vertices $w \in W^A$. There is an edge from v to a with capacity one if and only if v prefers a most (where $v \in V^A \cup W^A$). Each $a \in A$ yields an edge (a,y) with capacity $\bar{s}$.

We claim that the chair can make c a winner by adding ℓ_{AV} voters and deleting ℓ_{DV} voters according to the current guess if and only if there is an integral flow F with value $score_{(C,V)}(A) - \ell_{DV}^A + \ell_{AV}^A$ (that is, the greatest possible flow value) and minimum cost of at most $\lfloor \frac{n'}{2} \rfloor - score_{(C,\bar{V})}(d)$.

($\Rightarrow$): Suppose that the chair reaches his goal. Then there is a way to make c and d reach the runoff for some way to break ties, at least half of all voters in the final election prefer c to d, where the chair adds ℓ_{AV}^e voters approving of e ($e \in \{c,d\}$) and ℓ_{AV}^A voters from W^A, and he further deletes ℓ_{DV}^e voters voting for e ($e \in \{c,d\}$) and ℓ_{DV}^A voters from V^A. We construct the flow F as follows:

- Let $F((x,v^A)) = score_{(C,V)}(A) - \ell_{DV}^A$ and $F((x,w^A)) = \ell_{AV}^A$ as these are the numbers of voters in the final election that favor a candidate in A and belong to V and W, respectively.

- We let $F((v^A,v)) = 1$ for each voter $v \in V^A$ remaining in the election. If v is deleted from V, we have $F((v^A,v)) = 0$. Note that due to $F((x,v^A)) = score_{(C,V)}(A) - \ell_{DV}^A$, the same number of edges (v^A,v) have the flow value 1. More precisely, it holds $F((x,v^A)) = \sum_{v \in V^A} F((v^A,v))$.

- Moreover, we let $F((w^A,w)) = 1$ whenever voter $w \in W^A$ joins the election. Since exactly ℓ_{AV}^A such voters join the final election (accordingly, we have set $F((x,w^A)) = \ell_{AV}^A$), we have—following the flow conservation condition—exactly ℓ_{AV}^A edges (w^A,w) with flow value 1, that is, $\sum_{w \in W^A} F((w^A,w)) = F((x,w^A)) = \ell_{AV}^A$. For the remaining edges w (corresponding to voters in W^A not added to the election), we set $F((w^A,w)) = 0$.

- For each voter $v \in V^A \cup W^A$ belonging to the final election, we set $F((v,a)) = 1$, where $a \in A$ is voter v's most preferred candidate. If v does not belong to the final election, we set $F((v,a)) = 0$. Note that exactly $score_{(C,V)}(A) - \ell^A_{DV} + \ell^A_{AV}$ edges (i.e., the size of a maximum flow) (v,a) have the flow value 1.

- Observe that the total cost of edges (v^A, v) or (w^A, w) (where $v \in V^A$, $w \in W^A$) with flow value 1 is at most $\lfloor \frac{n'}{2} \rfloor - score_{(C,\bar{V})}(d)$ since otherwise a majority of voters in the final election would prefer d to c and c would lose against d in the final runoff.

- Finally, we set $F((a,y)) = \sum_{v \in V^A \cup W^A} F((v,a))$ for each $a \in A$. Observe that $F((a,y)) \leq \bar{s}$ as otherwise a would have more points than c or d and there would not exist any tie-breaking rule for which both c and d reach the runoff.

It follows that the flow F is feasible and has the required size.

($\Leftarrow$): Assume that a flow as described above exists. Accordingly, we construct an election where c and d reach the runoff for some way to break ties and c does not lose the pairwise comparison in the final election. Each edge $v \in V^A \cup W^A$ with $F((v,a)) = 1$ one-to-one corresponds to a voter v in the final election voting for $a \in A$. As the flow is integral, each edge (v,a) has either value one or zero. Observe that due to $F((x,v^A)) = score_{(C,V)}(A) - \ell^A_{DV}$ and $F((x,w^A)) = \ell^A_{AV}$, exactly $score_{(C,V)}(A) - \ell^A_{DV}$ voters in V^A and exactly ℓ^A_{AV} voters in W^A belong to the final election. As $F((a,y)) \leq \bar{s}$ for each $a \in A$, it follows that c and d are among the best two candidates (for some way to break ties). Since the total cost is at most $\lfloor \frac{n'}{2} \rfloor - score_{(C,\bar{V})}(d)$, at most this number of edges (v,a) has flow value 1. Note that according to our reasoning above, the final number of voters is exactly n' as there are exactly $score_{(C,\bar{V})}(e)$ voters favoring e ($e \in \{c,d\}$) in the final election. Moreover, the final number of voters favoring some $a \in A$ is equivalent to the value of the flow. As the total cost of the flow is small enough, at least half of all voters in the final election prefer c to d, and c is at least a tying winner in the resulting election.

In summary, our problem is in P as our algorithm runs only polynomially many guesses. There are $m-1$ pivotal non-distinguished candidates d, for each such candidate we have to check at most polynomially many guesses $(\ell^c_{AV}, \ell^d_{AV}, \ell^c_{DV}, \ell^d_{DV})$, and for some feasible guesses we compute a min-cost flow with a fixed flow value. Since both computing such a flow and the transformation from a guess to the corresponding min-cost flow problem are easy, our overall problem is in P. ❑

A similar algorithm can be applied for Veto with Runoff:

Theorem 5.2. VETO WITH RUNOFF-ECCAV+DV *is in* P.

Proof. Once more, our algorithm checks for each non-distinguished candidate d if both d and c can reach the runoff for some tie-breaking scheme and c beats or ties with d in the runoff. In case $|C| = 2$, the two voting rules Plurality with Runoff and Veto with Runoff coincide and we argue analogously to the proof of Theorem 5.1.

Let henceforth be $|C| > 2$ and d a pivotal candidate. We presume that the chair adds exactly ℓ_{AV} voters and deletes ℓ_{DV} voters. Suppose further that the chair adds ℓ^c_{AV} voters vetoing c, ℓ^d_{AV} voters vetoing d, and ℓ^A_{AV} voters vetoing candidates in $A := C \setminus \{c,d\}$. Moreover, the chair deletes ℓ^c_{DV} voters vetoing c, ℓ^d_{DV} voters vetoing d, and ℓ^A_{DV} voters vetoing candidates in A. It must hold $\ell_{AV} = \ell^c_{AV} + \ell^d_{AV} + \ell^A_{AV}$, $\ell_{DV} = \ell^c_{DV} + \ell^d_{DV} + \ell^A_{DV}$, and $\ell^e_X \in \mathbb{N}_0$ ($e \in \{A,c,d\}$, $X \in \{AV,DV\}$). Note that

the following conditions must be satisfied: $\ell^e_{AV} \leq \mathit{vetoes}_{(C,W)}(e)$ and $\ell^e_{DV} \leq \mathit{vetoes}_{(C,V)}(e)$ (where $e \in \{c,d,A\}$ and $\mathit{vetoes}_{(C,V)}(A) = \sum_{a \in A} \mathit{vetoes}_{(C,V)}(a)$; we analogously define $\mathit{vetoes}_{(C,W)}(A)$). Otherwise, we can immediately drop the sextuple $(\ell^c_{AV}, \ell^d_{AV}, \ell^A_{AV}, \ell^c_{DV}, \ell^d_{DV}, \ell^A_{DV})$ and proceed with the next one (if any).

It follows that the final veto numbers of c and d are uniquely determined for a given guess. We have $\mathit{vetoes}_{(C,\bar{V})}(e) = \mathit{vetoes}_{(C,V)}(e) - \ell^e_{DV} + \ell^e_{AV}$ for $e \in \{c,d\}$ in the final election $(C,\bar{V})$. Let $\bar{v} = \max(\mathit{vetoes}_{(C,\bar{V})}(c), \mathit{vetoes}_{(C,\bar{V})}(d))$. To ensure that both c and d reach the runoff for some way to break ties, every candidate $a \in A$ requires at least $\bar{v}$ vetoes in the final election. Hence, for the final election $(C,\bar{V})$ the following conditions must hold (we let V^A and W^A denote the voters in V and W vetoing a candidate in A, respectively). (1) Exactly $\mathit{vetoes}_{(C,V)}(A) - \ell^A_{DV}$ voters in V^A remain in the election, not being deleted by the chair, (2) exactly ℓ^A_{AV} voters are added from W^A, (3) for each $a \in A$ at least $\bar{v}$ voters in the final election veto a, and (4) at least half of all voters in the final election must prefer c over d. Since each voter vetoing c prefers d to c and the final election $(C,\bar{V})$ may include at most $\lfloor \frac{n'}{2} \rfloor$ voters preferring d to c (where $n' = |V| + \ell_{AV} - \ell_{DV}$ denotes the number of voters in the final election), there may be at most $\lfloor \frac{n'}{2} \rfloor - \mathit{vetoes}_{(C,\bar{V})}(c)$ voters in the final election that prefer d to c and veto a candidate in A. Note that in case this number is negative, we immediately discard the current sextuple as then c loses against d in the runoff, provided that both candidates reach the runoff at all. Notice that voters vetoing d never favor d over c.

To express these conditions, we define the following mincost flow problem. There is a source x, a sink y, and three vertices v^A, w^A, and r. Besides, each voter v in $V^A \cup W^A$ yields a vertex v. We further have a vertex for every $a \in A$. If not mentioned other, the cost of an edge is zero. The edges are defined as follows:

- There is an edge from x to v^A with capacity $\mathit{vetoes}_{(C,V)}(A) - \ell^A_{DV}$.
- Another edge goes from x to w^A with capacity ℓ^A_{AV}. These two edges assign vetoes to candidates in A in all votes of the final election. Observe that ℓ^A_{DV} voters from V^A are deleted and ℓ^A_{AV} voters are added from W^A.
- For each $v \in V^A$, there is an edge from v^A to v with capacity one. The edge cost is one if and only if the voter prefers d to c. Otherwise, the cost is zero.
- Analogously, we define edges from w^A to w (one for each voter $w \in W^A$).
- Each voter $v \in V^A \cup W^A$ yields the edges (v,a) and (v,r) with capacity 1 each, where a is the candidate vetoed by v.
- Each candidate a yields an edge from a to y with capacity $\bar{v}$.
- There is an edge (r,y) with capacity $(\mathit{vetoes}_{(C,V)}(A) - \ell^A_{DV} + \ell^A_{AV}) - (\bar{v} \cdot (|C| - 2)) = \mathit{vetoes}_{(C,\bar{V})}(A) - (\bar{v} \cdot |A|)$. (If this capacity is negative, we immediately discard our current guess as then the candidates in A cannot receive enough vetoes in total in order that each of them has $\bar{v}$ or more vetoes.)

Analogously to the proof of Theorem 5.1, we show that there is a successful control with the pivotal strategy sextuple and pivotal candidate d if and only if there is a flow with size $\mathit{vetoes}_{(C,V)}(A) -$

$\ell^A_{DV} + \ell^A_{AV}$ (that is, a flow of maximum size) and costs of at most $\lfloor \frac{n'}{2} \rfloor - \mathit{vetoes}_{(C,\bar{V})}(c)$ as then and only then each a has $\bar{v}$ or more vetoes (the flow assigns exactly $\bar{v}$ vetoes to candidates in A via the edges (a,y), the excessive vetoes for candidates in A are bypassed over the edges (v,r) and (r,y)). Note that the leftmost and the rightmost edges (i.e., (x,v^A) and (x,w^A) on the one hand and (a,y) and (r,y) on the other hand) carry the same total flow from source to sink. In case of a flow of value $\mathit{vetoes}_{(C,V)}(A) - \ell^A_{DV} + \ell^A_{AV}$, all vetoes for candidates in A in the final election are considered. Since each a has $\bar{v}$ or more vetoes, there is a tie-breaking method for which c and d go to the final stage. In case the costs are low enough, c does not lose the pairwise comparison with d in the runoff. Note that it is important to consider all voters vetoing candidates in A in the final election (i.e., not leaving out any veto positions for candidates in A in the final election) in order to check whether c really wins against or ties with d in the pairwise comparison against d. Possibly, fewer vetoes already suffice to assign $\bar{v}$ vetoes to every a, but we cannot say anything about the pairwise comparison between c and d when regarding fewer vetoes for candidates in A than there are in the final election.

Since there are $O(m)$ candidates d and $O(n^6)$ sextuples $(\ell^c_{AV}, \ell^d_{AV}, \ell^A_{AV}, \ell^c_{DV}, \ell^d_{DV}, \ell^A_{DV})$ to check, and computing an integral min-cost flow is easy, our overall problem is in P. ❑

Figure 5.1 provides an example of the flow algorithm defined in the proof of Theorem 5.2 where $\mathit{vetoes}_{(C,\bar{V})}(c) = \mathit{vetoes}_{(C,\bar{V})}(d) = 1$, that is, each candidate $a \in A = \{a_1, a_2\}$ requires at least one veto in the final election $(C,\bar{V})$ in order that c and d reach the runoff for some way to break ties. Suppose that $V^A = \{v_1, v_2\}$, $W^A = \{w_1, w_2\}$, $\ell^A_{AV} = 2$, and $\ell^A_{DV} = 1$. We further assume that both voters in V^A veto a_1, while both voters in W^A veto a_2. All voters in $V^A \cup W^A$ except v_1 prefer d to c. Observe that $|\bar{V}| = 5$ (three voters in $V^A \cup W^A$ and two voters vetoing c and d each). Hence, there may be at most one voter in $V^A \cup W^A$ preferring d to c because otherwise c would lose against d in the runoff. Notice that we can assign one veto to each candidate in A by assuming v_1 and w_1 to be in the final election. However, the vertex r ensures that all vetoes for candidates in A in the final election are taken into account. Hence, a flow must have size 3 (i.e., the total capacity of the rightmost edges). As each flow of size 3 costs at least two units, we reject for our guess.

Given the above results about exact multimode control, we obtain the following theorem.

Theorem 5.3. PLURALITY WITH RUNOFF-Y *and* VETO WITH RUNOFF-Y *are in* P *for all* $Y \in \{$*CCAV, CCDV, CCRV, CCAV+DV*$\}$ *and their exact counterparts.*

Proof. For ECCRV, we use the algorithms of Theorem 5.1 and Theorem 5.2 for $\ell_{AV} = \ell_{DV} =: \ell$. For CCRV, we regard $\ell + 1$ instances of ECCRV where the chair replaces exactly $\ell' \in \{0, 1 \ldots, \ell\}$ voters.

For ECCAV, we regard an instance of ECCAV+DV with $\ell_{DV} = 0$. In CCAV, we regard $\ell_{AV} + 1$ instances of ECCAV where the chair adds exactly $\ell'_{AV} \in \{0, 1, \ldots, \ell_{AV}\}$ voters.

ECCDV is a special case of ECCAV+DV with $\ell_{AV} = 0$. For CCDV, we simply have to consider at most $\ell_{DV} + 1$ ECCDV instances where the chair deletes exactly $\ell'_{DV} \in \{0, 1, \ldots, \ell_{DV}\}$ voters.

Finally, for inexact multimode control, we try out at most $(\ell_{AV} + 1) \cdot (\ell_{DV} + 1)$ exact multimode control problems where the chair adds precisely ℓ'_{AV} voters and deletes exactly ℓ'_{DV} voters, where $\ell'_{AV} \in \{0, \ldots, \ell_{AV}\}$ and $\ell'_{DV} \in \{0, \ldots, \ell_{DV}\}$.

Note that there are only polynomially many subproblems to regard for each case which all are in P. ❑

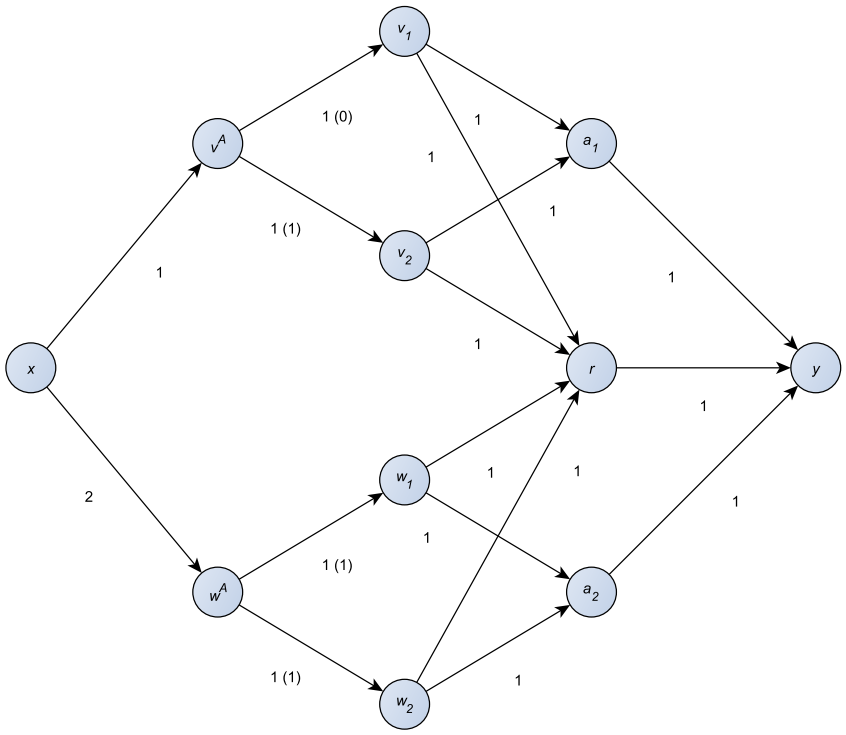

Figure 5.1: Example of the flow algorithm in the proof of Theorem 5.2

5.2.2 Bribery

In this section, we focus on bribery in Plurality with Runoff and Veto with Runoff. These results have not yet been published, but they fit well into our results about candidate and voter control. Our input is an election (C,V) with m candidates in candidate set C, n voters in voter set V (where $m \in \mathbb{N}$, $n \in \mathbb{N}_0$), a distinguished candidate c, and a nonnegative integer $\ell \leq |V|$—the bribery limit. W.l.o.g., we assume that $m \geq 2$ as otherwise c is trivially a winner being the only candidate.

Our first result reveals that bribery is easy for the rule Plurality with Runoff under parallel-universe tie-breaking.

Theorem 5.4. PLURALITY WITH RUNOFF-BRIBERY *is in* P

Proof. First assume that $|C| = 2$ ($C = \{c,d\}$). Then a rational briber bribes as many voters as possible voting for d and each of them approves of c after the bribery. It remains to check whether at least half of all voters vote for c after the bribery. We accept if and only if $score(c) + \ell \geq \frac{n}{2}$ holds. Otherwise, more voters in the final election prefer d to c than the other way around.

For $|C| > 2$, we define an algorithm based on a similar idea to the one for multimode control (cf. Theorem 5.1). We guess a candidate $d \neq c$ and check whether the briber can bribe w.l.o.g. [1] ℓ voters such that c and d reach the runoff for some tie-breaking rule and c wins or ties with d

[1] Even if fewer bribes suffice to make c win the election, this is no restriction as the additionally bribed voters may submit the same rankings as before.

in their head-to-head contest. We assume that the chair bribes λ_c voters voting for c, λ_d voters approving of d, and λ_A voters preferring a candidate in $A := C \setminus \{c,d\}$. Such a guess satisfies the condition $\ell = \lambda_c + \lambda_d + \lambda_A$ and $\lambda_c, \lambda_d, \lambda_A \in \mathbb{N}_0$. If $\lambda_c > score(c)$, $\lambda_d > score(d)$, or $\lambda_A > score(A) := \sum_{a \in A} score(a)$, we immediately discard the current guess.

Moreover, we assume that ℓ_c and ℓ_d bribed voters vote for c and d after the bribery, respectively. Note that it does not make sense when bribed voters favor other candidates. If c and d make it to the runoff, ℓ_c voters favor c and ℓ_d bribed voters prefer d, and there are still some bribes left (that is, $\ell_c + \ell_d < \ell$), a rational briber gives these overhanging points to c as this increases c's advantage compared with all other candidates and makes c win further pairwise comparisons against d which might turn out to be essential in the runoff stage. Thus, we regard all combinations $(\ell_c, \ell_d) \in \mathbb{N}_0^2$ with $\ell_c + \ell_d = \ell$.

In opposition—as we could observe above—the bribes may concern voters initially approving of c, d, or some candidate in A.

Notice that after the bribery, c and d have uniquely determined scores, that is, $score_{(C,\bar{V})}(e) = score_{(C,V)}(e) - \lambda_e + \ell_e$ ($e \in \{c,d\}$). Let $\bar{s} := \min(score_{(C,\bar{V})}(c), score_{(C,\bar{V})}(d))$. In order to get c and d into the runoff (for at least one way to break ties), the briber must ensure that each $a \in A$ has at most $\bar{s}$ points. Moreover, c must not lose the pairwise comparison against d in the runoff. Since d definitely wins each vote against c where d is ranked first, and as d may win at most $\lfloor \frac{n}{2} \rfloor$ pairwise comparisons against c in total, the final election may contain at most $\lfloor \frac{n}{2} \rfloor - score_{(C,\bar{V})}(d)$ voters favoring a candidate in A and preferring d to c. Note that in case this number is negative, we reject for the current guess as a majority of voters favor d and accordingly d wins the pairwise comparison against c. We define the following flow network. There is a source x, a sink y, a vertex v for each voter v approving of some $a \in A$ (once more, let us call the set of these voters V^A), and a vertex a for each $a \in A$. The edges are defined as follows.

- Each voter $v \in V^A$ yields an edge (x,v) with capacity one.
- For each $v \in V^A$, $a \in A$, there is an edge (v,a) with capacity one if and only if v approves of a. The cost of the edge is one if and only if v prefers d to c. Otherwise, the cost is zero.
- Each $a \in A$ yields an edge (a,y) with capacity $\bar{s}$.
- All costs are zero if not mentioned other.

Similarly to the proof of Theorem 5.1, we accept for the current guess if and only if the flow network yields an integral flow of the size $score(A) - \lambda_A$ and total cost of at most $\lfloor \frac{n}{2} \rfloor - score_{(C,\bar{V})}(d)$ as then and only then c and d make it to the runoff for some way to break ties and c does not lose the pairwise comparison against d.

Our problem is surely in P since the number of guesses to check is polynomially bounded and our problem polynomially reduces to computing an integral flow with minimum cost. ❑

For Veto with Runoff, the bribery problem turns out to be easy, too. Again, the proof makes use of computing a flow of minimum cost.

Theorem 5.5. VETO WITH RUNOFF-BRIBERY *is in* P.

Proof. First of all, for $|C| = 2$, the briber bribes as many voters vetoing c as possible and each of them vetoes the other candidate after the bribery. It remains to check whether c has no more vetoes than the other candidate in the resulting election.

Now let $|C| > 2$. Similarly to the proof of Theorem 5.2, our algorithm guesses a candidate $d \neq c$ and we check whether c and d can make it to the runoff and d does not win the pairwise duel against c. We further test whether this is possible when the briber bribes λ_c voters vetoing c, λ_d voters vetoing d, and λ_A voters vetoing some candidate $a \in A := C \setminus \{c,d\}$. Moreover, we suppose that ℓ_d bribed voters veto d after the bribery, whereas ℓ_A bribed voters give their veto to some $a \in A$. We may assume that no bribed voter vetoes c after the bribery since this makes c lose his pairwise comparison against d. Therefore, we suppose that either d or some $a \in A$ get these vetoes. Notice that for a given guess it holds $\ell_A + \ell_d = \ell$, $\lambda_A + \lambda_d + \lambda_c = \ell$, and $\lambda_d, \lambda_c, \lambda_A, \ell_d, \ell_A \in \mathbb{N}_0$.

Our current guess must additionally satisfy the conditions $\lambda_c \leq \mathit{vetoes}(c)$, $\lambda_d \leq \mathit{vetoes}(d)$, and $\lambda_A \leq \mathit{vetoes}(A) := \sum_{a \in A} \mathit{vetoes}(a)$. Otherwise, we discard this guess and check the next one.

Depending on a guess (that is, a combination $(d, \lambda_c, \lambda_d, \lambda_A, \ell_d, \ell_A)$), we obtain uniquely determined veto numbers $\mathit{vetoes}_{(C,\bar{V})}(c) = \mathit{vetoes}_{(C,V)}(c) - \lambda_c$ and $\mathit{vetoes}_{(C,\bar{V})}(d) = \mathit{vetoes}_{(C,V)}(d) - \lambda_d + \ell_d$. In order to get both c and d into the runoff (for some tie-breaking scheme), it must hold $\mathit{vetoes}_{(C,\bar{V})}(a) \geq \max(\mathit{vetoes}_{(C,\bar{V})}(c), \mathit{vetoes}_{(C,\bar{V})}(d)) =: \bar{v}$ for each $a \in A$ in the final election $(C, \bar{V})$.

Moreover, c must defeat d or tie with d in their head-to-head contest after the bribery. As each voter vetoing c prefers d to c, the distinguished candidate c may lose at most $\lfloor \frac{n}{2} \rfloor - \mathit{vetoes}_{(C,\bar{V})}(c)$ pairwise comparisons in votes of the final election where some $a \in A$ is vetoed. Observe that if $\lfloor \frac{n}{2} \rfloor - \mathit{vetoes}_{(C,\bar{V})}(c) < 0$, we discard the current guess as a proper majority of voters favor d over c and—provided that both c and d make it to the runoff stage—c is not a runoff winner.

Hence, we check whether it is possible to find $\mathit{vetoes}(A) - \lambda_A$ voters vetoing some $a \in A$ left unchanged by the briber and at most $\lfloor \frac{n}{2} \rfloor - \mathit{vetoes}_{(C,\bar{V})}(c)$ of them prefer d to c. Note that ℓ_A voters veto candidates in A after the bribery (no matter whom they have vetoed before the bribery) and their vetoes can be arbitrarily set by the briber. Each of these voters is supposed to prefer c to d without any restriction as they either veto d (and hence c is trivially ranked better) or they veto a candidate in A and thus c and d are w.l.o.g. on the first and second position in these votes, respectively. Moreover, each $a \in A$ must have $\bar{v}$ or more vetoes in the final election. All these preconsiderations lead us to the following mincost flow problem. The flow network contains a source x, a sink y, vertices v^A, b, r, a vertex v for each voter v vetoing a candidate in A (we call the set of these voters V^A), and a vertex a for each candidate $a \in A$. If not mentioned other, the cost of an edge is zero. The edges are defined as follows.

- There is an edge (x, v^A) with capacity $\mathit{vetoes}(A) - \lambda_A$. This edge is to ensure that exactly $\mathit{vetoes}(A) - \lambda_A$ voters in V^A remain unchanged.
- There is an edge (x, b) with capacity ℓ_A. This edge makes sure that precisely ℓ_A bribed voters veto candidates in A after the bribery.
- From v^A, there is an edge to each $v \in V^A$ with capacity 1. The edge cost is 1 if and only if v prefers d to c.
- For each $v \in V^A$, there are two edges (v, a) and (v, r) with capacity 1 each, where a is the candidate vetoed by v.

- For each $a \in A$, there is an edge (b,a) with capacity ℓ_A. Moreover, there is an edge (b,r) with capacity ℓ_A.
- For every $a \in A$, there is an edge (a,y) with capacity $\bar{v}$.
- There is an edge (r,y) with capacity $\mathit{vetoes}(A) - \lambda_A + \ell_A - (m-2)\bar{v}$. (Again, this number must not be negative as then there are not enough vetoes in total to distribute such that each $a \in A$ can get (exactly) $\bar{v}$ vetoes. Thus, c and d do not make it to the runoff together. We immediately discard our current guess in this case.)

Similarly to the proofs of Theorem 5.1, 5.2, and 5.4, it follows that the briber can make c a runoff winner against d under the current guess if and only if the flow network yields an integral flow with flow value $\mathit{vetoes}(A) - \lambda_A + \ell_A$ and a total cost of at most $\lfloor \frac{n}{2} \rfloor - \mathit{vetoes}_{(C,\bar{V})}(c)$.

The basic ideas are as follows. The edges starting from the source assign $\mathit{vetoes}(A) - \lambda_A$ vetoes to (unchanged) voters in V^A and ℓ_A vetoes that bribed voters assign to candidates in A. Hence, the number of vetoes for candidates in A in the final election is equivalent to the flow size. Again, each voter in V^A preferring d to c in the final election, according to the flow, costs one unit. Vertex b feasibly distributes ℓ_A vetoes to candidates in A. The vertex r functions as an auxiliary vertex that bypasses the excessive vetoes for the $a \in A$ to the target vertex. Each a requires at least $\bar{v}$ vetoes. If possible, the flow network ensures that exactly $\bar{v}$ vetoes are considered (via the edges (v,a) and (a,y)) and the remaining, excessive vetoes (that would lead to a higher veto number for $a \in A$) flow to y via vertex r. [2] Observe that $(\sum_{a \in A} \mathit{cap}((a,y))) + \mathit{cap}((r,y)) = \mathit{cap}((x,v^A)) + \mathit{cap}((x,b))$. The values on both sides of the equation correspond to the size of a maximum flow. ❑

Notice that both proofs hold for the exact and inexact variant (we assumed that the briber meets his bribery limit). The reason is that—in contrast to control—the briber can *effectively* bribe fewer voters when some bribed voters approve or veto the same candidate(s) as before. In control by deleting voters, for example, we cannot say that the chair faces the same problem when deleting five voters instead of three voters. Assume, e.g., that two deleted voters approve of c, which may hurt c decisively. In bribery, bribing a voter approving of c can be made undone by making the voter approve of c again.

5.2.3 Candidate Control

Consider now candidate control. We show that candidate control is generally NP-complete for both runoff rules. In each proof, we reduce from RX3C and are given an instance $(B,\mathscr{S})$ with $B = \{b_1,\ldots,b_{3m}\}$, $\mathscr{S} = \{S_1,\ldots,S_{n=3m}\}$, $S_i \subset B$, and $|S_i| = 3$, for each i, $1 \leq i \leq n$. Sometimes, we rename the candidates in B for practical reasons. For instance, we often write $S_i = \{b_1^i, b_2^i, b_3^i\}$ to emphasize the three candidates in B belonging to a particular 3-set S_i. We start with CCAC for Plurality with Runoff.

[2] Again, we included the r vertex into our construction as thus we can decide whether c still beats or ties with d in the head-to-head contest when we consider all voters vetoing candidates in A in the final election. If we did without vertex r and its incident edges, our flow network would show whether it is possible that c and d make it to the runoff for some tie-breaking method, but then a statement whether c wins or loses against d (or ties with d) would not be possible in general.

Theorem 5.6. PLURALITY WITH RUNOFF-CCAC *is* NP*-complete.*

Proof. To show our result, we give a reduction from the RX3C problem to PLURALITY WITH RUNOFF-CCAC. Let $(B,\mathscr{S})$ be an instance of RX3C, i.e., each $b_j \in B$ occurs in exactly three subsets in $\mathscr{S}$. Remind that it holds $|\mathscr{S}| = |B| = 3m$ for any instance of RX3C. Our CCAC instance is defined as follows. The set of registered candidates is $C = \{c,q\} \dot\cup B$, whereas $D = \mathscr{S}$ contains the unregistered candidates. c is our designated candidate. We create the following $15+24m$ votes:

1. There are eight voters ranking q first and all other candidates in $C \cup D$ behind.
2. Seven voters rank c first and all other candidates in $C \cup D$ behind.
3. For each $b_j \in B$, there are two voters voting $b_j \succ c \succ ((C \cup D) \setminus \{b_j, c\})$.
4. Finally, for each $S_i \in \mathscr{S}$, $S_i = \{b_1^i, b_2^i, b_3^i\}$, we create two votes with preference $S_i \succ b_j^i \succ \cdots \succ c \succ q$, $1 \le j \le 3$ (how the candidates in between are ranked, is immaterial to our analysis). In other words, each S_i yields six voters in this group, two for each b_j^i $(1 \le j \le 3)$.

We are allowed to add at most m candidates. This completes the construction. Observe that in the original election, we obtain $score(q) = 8$, $score(c) = 7$, and $score(b_j) = 8$ for every $j \in [3m]$. Thence, c is not in the runoff for any tie-breaking rule, and consequently c is not a Plurality with Runoff winner.

We claim that an exact cover $\mathscr{S}'$ of B exists if and only if at most m candidates can be added from D such that c is a winner.

($\Rightarrow$): Assume that there is an exact 3-set cover $\mathscr{S}'$. Notice that after adding candidates in $\mathscr{S}'$, q has 8 points, c has 7 points, every $S_i \in \mathscr{S}'$ has 6 points, and every $b_j \in B$ has 8-2=6 points. Hence, q and c go into the runoff stage. As $N(c,q) = 24m+7 > \frac{24m+15}{2} = \frac{|V|}{2}$ (this inequality is valid for every $m \in \mathbb{N}$), more voters prefer c to q than vice versa and consequently c is a unique runoff winner against q.

($\Leftarrow$): Presume that the chair can make c a winner by adding no more than m candidates. Observe that to ensure c to survive the first round of the election, at least m candidates must be added. Otherwise, there is at least one candidate $b_j \in B$ who receives 8 approvals, and q and this candidate have more points than c. This means that c does not reach the runoff. Let $\mathscr{S}'$ be a solution. As discussed, we have $|\mathscr{S}'| = m$. If $\mathscr{S}'$ is not an exact 3-set cover, again there is a candidate $b_j \in B$ not belonging to any subset of $\mathscr{S}'$ and thus receiving 8 approvals after the addition of the candidates in $\mathscr{S}'$ to the election. Once more, c does not reach the runoff stage. Therefore, $\mathscr{S}'$ must be an exact 3-set cover. ❑

We achieve another hardness result for the Veto with Runoff rule.

Theorem 5.7. VETO WITH RUNOFF-CCAC *is* NP*-complete.*

Proof. We prove the claim by a reduction from the RX3C problem. For a given RX3C instance $(B,\mathscr{S})$, we create the following VETO WITH RUNOFF-CCAC instance.

Let $C = B \dot\cup \{c,q\}$ be the set of registered candidates, $\mathscr{S}$ be the set of unregistered candidates (for each $S_i \in \mathscr{S}$ we create a candidate denoted by the same symbol), and c the distinguished

candidate. Let V be the set of voters consisting of the following $18m^2+6mn+11n-9m+1 = 36m^2+24m+1$ voters (the equality holds because of $n=3m$; when defining the voters, we yet distinguish between the indexes n and m in order to emphasize which voter groups are based on the S_i and which ones refer to the elements in B):

1. There is a voter of the form $\mathscr{S} \succ B \succ c \succ q$,
2. for each i, $1 \leq i \leq n$, there are $3m+1$ voters of the form $q \succ B \succ c \succ \mathscr{S} \setminus \{S_i\} \succ S_i$,
3. for each i, $1 \leq i \leq n$, there are $3m+1$ voters of the form $c \succ B \succ q \succ \mathscr{S} \setminus \{S_i\} \succ S_i$,
4. for each j, $1 \leq j \leq 3m$, there are $6m-3$ voters of the form $c \succ q \succ \mathscr{S} \succ B \setminus \{b_j\} \succ b_j$,
5. for each i, $1 \leq i \leq n$, there are three voters of the form $q \succ B \succ \mathscr{S} \setminus \{S_i\} \succ c \succ S_i$,
6. for each i, $1 \leq i \leq n$, with $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, there are two voters of the form $c \succ q \succ B \setminus \{b_1^i\} \succ \mathscr{S} \setminus \{S_i\} \succ b_1^i \succ S_i$,
7. for each i, $1 \leq i \leq n$, with $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, there are two voters of the form $c \succ q \succ B \setminus \{b_2^i\} \succ \mathscr{S} \setminus \{S_i\} \succ b_2^i \succ S_i$, and
8. for each i, $1 \leq i \leq n$, with $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, there are two voters of the form $c \succ q \succ B \setminus \{b_3^i\} \succ \mathscr{S} \setminus \{S_i\} \succ b_3^i \succ S_i$.

We are allowed to add at most m candidates, i.e., $\ell = m$. Note that c has $3m(3m+1)+9m$ vetoes (because of vetoes from voter groups 2. and 5.), q has $3m(3m+1)+1$ vetoes (due to groups 1. and 3.), and for every j, $1 \leq j \leq 3m$, b_j has $6m+3$ vetoes (from the remaining four voter groups). In particular, it holds $\mathit{vetoes}(b_j) = 6m+3 < \mathit{vetoes}(c) = 3m(3m+1)+9m$ which is equivalent to $9m^2+6m-3>0$. This implies $9m^2+6m-3 \geq 9m+6m-3 = 15m-3 \geq 15-3 = 12 > 0$ for every natural number m. Since $|B| \geq 3$, the designated candidate c does not even reach the runoff and is thus not a Veto with Runoff winner. We point out that as soon as one candidate in $\mathscr{S}$ is added, q's number of vetoes collapses to one.

We claim that the chair can make c a Veto with Runoff winner by adding at most m candidates if and only if an exact cover of B exists.

($\Leftarrow$): Assume that there is an exact 3-set cover $\mathscr{S}'$ of B. After adding the candidates in $\mathscr{S}'$, candidate q has 1 veto, for every i, $1 \leq i \leq n$, each $S_i \in \mathscr{S}'$ has at least $(3m+1)+(3m+1) = 6m+2$ vetoes (due to the second and third voter group), for every j, $1 \leq j \leq 3m$, b_j has $6m+3-2 = 6m+1$ vetoes, and c has $6m$ vetoes. Hence, q and c move to the runoff stage. It remains to show that c does not lose the pairwise comparison against q. More precisely, we even prove that c wins the head-to-head contest against q. Observe that all voters except from groups two and five prefer c to q. Thus, there are exactly $3m(3m+1)+3m \cdot 3$ voters in total who prefer q to c. Consequently—considering that there are $36m^2+24m+1$ voters in V—a total of $27m^2+12m+1$ voters like c more than q. We obtain $N(c,q) = 27m^2+12m+1 > \frac{36m^2+24m+1}{2} = \frac{|V|}{2}$ which is equivalent to $18m^2+1>0$ (which holds for every $m \in \mathbb{N}$). Consequently, a majority of voters prefer c to q.

($\Rightarrow$): Assume that adding $\mathscr{S}' \subseteq \mathscr{S}$ from the set of unregistered candidates to the election makes c a winner under Veto with Runoff.

Observe first that $\mathscr{S}'$ must contain exactly m candidates since otherwise, due to voter group five, c would have at least $3(2m+1) = 6m+3$ vetoes. On the other hand, at least one candidate in B would have at most $6m+3-2 = 6m+1$ vetoes as at least one S_i is added (for $\mathscr{S}' = \emptyset$, we already know that c is not a winner). Hence, this candidate and q would both have fewer vetoes than c, and c would not even reach the runoff stage. So, we have $|\mathscr{S}'| = m$. It follows that c has $6m$ vetoes after the addition of m candidates in $\mathscr{S}'$. If $\mathscr{S}'$ is not an exact 3-set cover, there must be a candidate $b_j \in B$ occurring in at least two subsets of $\mathscr{S}'$. Then, because of voter groups 6-8, this candidate has at most $6m+3-4 = 6m-1$ vetoes and this veto number is smaller than the veto number of c. Since q has fewer vetoes than c as well, c does not reach the runoff stage. So, we can conclude that $\mathscr{S}'$ is an exact 3-set cover. ❑

We can also settle a hardness result for Plurality with Runoff and CCDC.

Theorem 5.8. PLURALITY WITH RUNOFF-CCDC *is* NP*-complete.*

Proof. We prove the claim by a reduction from the RX3C problem. For a given RX3C instance $(B, \mathscr{S})$, we create the following instance of PLURALITY WITH RUNOFF-CCDC. Let $C = \{c, q\} \dot\cup B \dot\cup \mathscr{S}$ be the set of candidates and c the distinguished candidate. Let V be the set of voters consisting of the following $9m^2 + 21m + 1$ voters (recall that $n = 3m$ holds).

1. There are $2m$ voters of the form $q \succ b_1 \succ b_2 \succ \cdots \succ b_{3m} \succ \mathscr{S} \succ c$,
2. there are $m+1$ voters of the form $q \succ b_{3m} \succ b_{3m-1} \succ \cdots \succ b_1 \succ \mathscr{S} \succ c$,
3. for each j, $1 \le j \le 3m$, there are $3m-3$ voters of the form $b_j \succ B \setminus \{b_j\} \succ \mathscr{S} \succ c \succ q$,
4. for each i, $1 \le i \le n$, there are three voters of the form $S_i \succ c \succ C \setminus \{S_i, c, q\} \succ q$,
5. for each i, $1 \le i \le n$, with $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, there are two voters of the form $S_i \succ b_1^i \succ C \setminus \{S_i, c, q, b_1^i\} \succ c \succ q$,
6. for each i, $1 \le i \le n$, with $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, there are two voters of the form $S_i \succ b_2^i \succ C \setminus \{S_i, c, q, b_2^i\} \succ c \succ q$, and
7. for each i, $1 \le i \le n$, with $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, there are two voters of the form $S_i \succ b_3^i \succ C \setminus \{S_i, c, q, b_3^i\} \succ c \succ q$.

Furthermore, assume that the chair may delete m candidates from C. W.l.o.g., we require that $m \ge 4$. In the original election, c has no point at all, whereas all other candidates have some points. Thus, c certainly does not reach the runoff and cannot be a winner under the Plurality with Runoff rule.

We claim that c can be made a Plurality with Runoff winner by deleting at most m candidates if and only if an exact cover of B exists.

($\Leftarrow$): Assume there is an exact 3-set cover $\mathscr{S}'$. After deleting the candidates in $\mathscr{S}'$, q has $2m + m + 1 = 3m + 1$ approvals, c has $3m$ approvals, every $S_i \in \mathscr{S} \setminus \mathscr{S}'$ has 9 approvals, and for every j, $1 \le j \le 3m$, b_j has $3m - 3 + 2 = 3m - 1$ approvals. Therefore—not least because $m \ge 4$ holds—q and c go to the runoff stage, implicating that c is the final winner as $N(c,q) =$

$9m^2+18m > \frac{|V|}{2} = \frac{9m^2+21m+1}{2}$ (which is equivalent to $9m^2+15m-1>0$; this in turn follows from $9m^2+15m-1 \geq 9+15-1 = 23 > 0$) holds for every $m \in \mathbb{N}$, and in particular for $m \geq 4$. In other words, a majority of voters favor c over q. Note that all voters except from the first two voter groups like c more than q.

($\Rightarrow$): Presume now that the chair can make c a Plurality with Runoff winner of the election by deleting at most m candidates. Let us denote these candidates by C'. Note that $q \notin C'$ since otherwise there would be two candidates in B receiving at least $3m-3+2m = 5m-3$ and $3m-3+m+1 = 4m-2$ approvals, preventing c from winning. [3] Furthermore, none of the candidates in B should be deleted, i.e., $B \cap C' = \emptyset$. In fact, if we delete a candidate $b_j \in B$, then the candidate ranked immediately after b_j in the $3m-3$ votes created for b_j (group three) would receive at least $(3m-3)+(3m-3) = 6m-6$ approvals, preventing c from going to the runoff stage. This means that the deletion of one candidate in B invites the deletion of all candidates in B, to make c the winner. However, we are allowed to delete at most m candidates. Therefore, we have $C' \subseteq \mathscr{S}$. After deleting the candidates in C', c has $3|C'|$ approvals, due to the fourth voter group. Note that $|C'| = m$ must hold. Otherwise, at least one candidate in B would receive more approvals than candidate c after the deletion of all candidates in C' as c would have at most $3(m-1)$ approvals and some b_j would have at least $3m-3+2 = 3m-1$ points in the final election, due to voter groups 3 and 5-7. Thence, c would not go to the runoff stage. Therefore, we know that c receives $3m$ approvals after the deletion of all candidates in C'. If C' is not an exact 3-set cover, then there must be a candidate $b_j \in B$ who occurs in at least two subsets S_i belonging to C'. Due to the construction, this candidate receives at least $3m-3+2+2 = 3m+1$ approvals ($3m-3$ approvals from the third voter group and at least four approvals according to two S_i sets containing b_j), implying that q and this candidate have more points than c, and c does not reach the runoff. Thus, the sets S_i in C' must form an exact 3-set cover of B. ❑

We can settle a hardness result for Veto with Runoff as well.

Theorem 5.9. VETO WITH RUNOFF-CCDC *is* NP*-complete.*

Proof. We prove the theorem reducing once again from the RX3C problem. For an instance $(B, \mathscr{S})$ of RX3C, we create the following instance of CCDC. We are given the candidate set $C = \{c,q\} \dot\cup B \dot\cup \mathscr{S}$. c is our distinguished candidate. There are $6n+1 = 18m+1$ (as $n = 3m$ holds for RX3C instances) voters defined as follows:

1. There is a voter with preference $\mathscr{S} \succ B \succ q \succ c$.
2. For each $S_i = \{b_1^i, b_2^i, b_3^i\} \in \mathscr{S}$, we create six votes as follows (number of votes: preferences of the votes):

   $2 : c \succ q \succ B \setminus \{b_1^i\} \succ \mathscr{S} \setminus \{S_i\} \succ b_1^i \succ S_i$,

[3] Note that if we delete q (and possibly some b_j), we leave at least $|B| - (m-1) = 2m+1$ candidates from B in the election as the chair may delete at most m candidates in total and one deletion is reserved for q. Let b' denote the candidate b_j in the final election with smallest index j, b'' denotes the one with largest index, i.e., formally there are two candidates $b_{j_1} =: b'$ and $b_{j_2} =: b''$ (with $1 \leq j_1 < j_2 \leq 3m$) not deleted, and all candidates $b_1, \ldots, b_{j_1-1}$ and $b_{j_2+1}, \ldots, b_{3m}$ are deleted by the chair. It follows that b' has at least $5m-3$ points (from group 1 and 3), and b'' has at least $4m-2$ approvals from voters in group 2 and 3. Observe that c gets three points for each deleted S_i. As at most $m-1$ such candidates are deleted, c has no more than $3(m-1)$ points and this score is below the score of both b' and b''.

$2 : c \succ q \succ B \setminus \{b_2^i\} \succ \mathscr{S} \setminus \{S_i\} \succ b_2^i \succ S_i$, and
$2 : c \succ q \succ B \setminus \{b_3^i\} \succ \mathscr{S} \setminus \{S_i\} \succ b_3^i \succ S_i$.

We are allowed to delete at most m candidates.

Observe that every $b_j \in B$ and q have 0 vetoes, c has 1 veto, and every $S_i \in \mathscr{S}$ has 6 vetoes. Hence, the current winner must be from $B \cup \{q\}$ since two of these candidates reach the runoff stage, depending on the tie-breaking rule used.

We claim that there is an exact cover $\mathscr{S}'$ of B if and only if the chair can make c a winner by deleting at most m candidates.

($\Rightarrow$): Let $\mathscr{S}'$ be an exact 3-set cover of $(B, \mathscr{S})$. Then, after deleting the candidates in $\mathscr{S}'$, c has 1 veto, every b_j has 2 vetoes, q has 0 vetoes, and every $S_i \in \mathscr{S} \setminus \mathscr{S}'$ (i.e., the candidates S_i not deleted) has 6 vetoes. Hence, c and q survive the first stage of voting. As all except the first voter prefer c to q (and there are at least three voters in total), c becomes the unique winner.

($\Leftarrow$): Suppose now that the chair can make c a winner by deleting the candidate subset C', where $|C'| \leq m$. We first argue that it must hold $|C'| = m$ and $C' \subseteq \mathscr{S}$ since otherwise at least two non-distinguished candidates with zero vetoes remain in the election (no matter whether q and/or candidates in B are among the candidates deleted). If the sets in C' do not exactly cover B, there are candidates $b_j \in B$ who do not occur in any subset of C'. Such candidates would have 0 vetoes. Hence, c would not even survive the first stage of voting in this case. So, we can conclude that all sets in C' must exactly cover B. ❑

Note that all hardness results hold regardless of the tie-breaking rule used. It remains to show the replacing candidates cases. Before continuing, we provide the following definition:

Definition 5.10. *[122, 123] A voting rule satisfies the condition* Insensitiveness to Bottom-ranked Candidates (IBC) *if the winners of a given election do not change after a proper subset of candidates who are ranked after all other candidates in all votes are deleted.*

This definition turns out to be useful in the following lemma.

Lemma 5.11. *[122, 123] Let $\mathscr{F}$ be a voting rule satisfying IBC. Then, $\mathscr{F}$-*CCRC *is* NP*-hard if $\mathscr{F}$-*CCDC *is* NP*-hard.*

Now we are equipped to determine the complexity for constructive control by replacing candidates for our two runoff rules.

Theorem 5.12. PLURALITY WITH RUNOFF-CCRC *and* VETO WITH RUNOFF-CCRC *are* NP*-complete.*

Proof. The hardness result for Plurality with Runoff follows from Lemma 5.11, Theorem 5.8, and the fact that Plurality with Runoff fulfills the criterion IBC.

For Veto with Runoff, we adjust the instance in the proof of Theorem 5.7 (CCAC) as follows: Introduce m additional candidates $d_1, \ldots, d_m$ ranked first by every voter, that is, each voter v votes $d_1 \succ d_2 \succ \ldots \succ d_m \succ \overrightarrow{R_v}$, where $\overrightarrow{R_v}$ is the original ranking of v in the proof of Theorem 5.7. As the chair may exchange m candidates, these m candidates are deleted with highest priority. In case two or more d_j remain in the election, two d_j reach the runoff. If one d_j remains in the election, c cannot make it to the runoff either (compare the proof of Theorem 5.7; recall that q has only one veto when

Voting rule	Bribery	AV	DV	RV	AV+DV	AC	DC	RC	AC+DC
Plurality with Runoff	P	P	P	P	P	NPC	NPC	NPC	NPC
Veto with Runoff	P	P	P	P	P	NPC	NPC	NPC	NPC

Table 5.1: Results for bribery and control in Plurality with Runoff and Veto with Runoff. When referring to control, we use a two- instead of four-letter code and drop the two letters CC. E.g., we write "AV" instead of "CCAV". The complexities are the same when we regard the exact versions of the problems instead. All polynomial-time results hold under parallel-universe tie-breaking. All hardness results hold regardless of tie-breaking rules and both under the co- and unique-winner model in the runoff. Key: P stands for "polynomial-time solvable", NPC stands for "NP-complete".

at least one S_i is added and hence—in case some d_j is not deleted—our distinguished candidate is at most third best candidate in the preliminary round). Hence, all d_j must be replaced with the m candidates added in the proof of Theorem 5.7. ❑

Note that the hardness result for all exact variants follow as well, with the same limits as in their inexact variants. Moreover, hardness follows for exact and inexact multimode control (allowing a chair to add and delete candidates and voters).

5.3 Conclusion

We have investigated the computational complexity of bribery, candidate control, and voter control for Plurality with Runoff and Veto with Runoff in their constructive variants, closing the gaps in the literature. Table 5.1 summarizes our results for the two runoff rules.

Contrary to the previous chapters, we have only provided the results under full information. While investigating replacement control for various voting rules such as Maximin, Copeland, or Condorcet (cf. our results in [73]), we have noticed that—to the best of our knowledge—neither bribery nor control has been solved for these two runoff rules up to then. This appeared to us somewhat surprising because both voting rules are frequently used in practice, in their pure form as well as in variations such as multi-stage versions or as adaptions allowing revoting in the runoff stage. We point out that studying scoring rules in a runoff version fits in the line of investigating two-stage rules as we have done in Chapter 4 in the context of lot-based voting rules. In contrast to lotteries selecting a subset of voters and applying a voting rule to these voters, runoff rules rule out all but the two best candidates and apply the Weak Condorcet or Condorcet rule to these two candidates—depending on the winner model. We have decided to present only the results for Plurality with Runoff and Veto with Runoff and ignore the other results from [73] as the focus in this thesis lies on k-Approval and k-Veto. Studying k-Approval/-Veto, $k \geq 2$, in a runoff version appears to be a promising task for future research.

Our findings in this chapter state that the worst-case complexities for Plurality with Runoff and Veto with Runoff match the complexities for their counterparts without runoff (for the results for Plurality and Veto without a runoff, we refer to [79] for bribery, [16] for control by adding/deleting voters/candidates, [81] for multimode control, and [122] for replacement control). More precisely, constructive control by adding, deleting, and replacing candidates is hard for both runoff rules. These results hold regardless of the given tie-breaking rule, for multimode control, and for the ex-

act variants of multimode control and control by adding, deleting, and replacing candidates. Both bribery and control by adding, deleting, and replacing voters are easy for both runoff rules. By providing the polynomial-time results for the most general voter control problem *exact multimode control*, the other voter control results follow. Although the problems are not harder than the corresponding problems without runoff, we have to point out several aspects. Firstly, the proofs are surely more involved. E.g., bribery in Plurality and Veto (under full information) can be solved by means of simple greedy algorithms [79], but the proofs for the runoff versions require more complex constructions with transformations of some subproblems to the polynomial-time solvable INTEGRAL MIN-COST FLOW problem. Secondly, all polynomial-time results have been shown under the parallel-universe tie-breaking rule. The question arises whether the problem is easy for other or even for arbitrary tie-breaking rules. Thirdly, the results hold under the assumption that each voter provides a complete ranking, revoting is not allowed, and each voter shows up to the runoff election. This poses some further intriguing questions. One could study what happens when the voters' votes are partial according to a model $X \in \text{PIM}$, a lot-based mechanism determines the voters whose votes count or the voters who actually participate in the runoff election, or how the results change when some or all voters may change their ballots in the final runoff. Moreover, variants of these two runoff rules, such as multi-stage versions (the multi-stage version of Plurality with Runoff is called STV in literature) or other variations raise further questions. Finally, we have to mention destructive control, which has been studied by us in [73], and destructive bribery. Nevertheless, we have decided to omit these results as we merely concentrate on constructive problems in Part I. The proofs of the destructive versions basically use the same constructions as the constructive versions. For the easy problems, it suffices to guess a non-distinguished candidate that is a runoff winner for at least one way to break ties, while the constructions showing hardness of constructive candidate control can be adjusted to prove the corresponding results for destructive candidate control.

In this chapter, we have considered exact control where the chair must meet his adding, deleting, and/or replacing limits. By studying exact multimode voter control, we could prove membership in P for the subproblems (E)CCAV, (E)CCDV, (E)CCAV+DV, and (E)CCRV at the same time. Although exact control has been widely ignored in literature so far, to the best of our knowledge, one can fancy many real-world applications where an election has to consist of fixed numbers of candidates or voters. One such application is given by lot-based voting (cf. Chapter 4) which can be regarded as an exact version of CCAV with an empty set of registered voters, V being the set of unregistered voters, and the lottery adds exactly K voters. Observe that the complexity might go up when we move from CCAV to ECCAV. By way of example, we know that 3-APPROVAL-CCAV is in P [119]. In contrast, 3-APPROVAL-ECCAV is NP-complete. We may simply borrow the reduction showing that LOTTHEN3-APPROVAL-POSSIBLE WINNER is NP-complete (see also Theorem 6.1 in [58] and Chapter 4 about lot-based voting). The question arises whether there are other instances where the exact variant of the given problem is hard and the inexact variant is easy.

As pointed out above, our original motivation behind our work in and around [73] was to obtain a complete picture over the complexities for replacement control. During our research, we could observe that (1) many other control problems had been open up to then and (2) control and bribery in two basic voting rules—Plurality with Runoff and Veto with Runoff—had been ignored in literature by then (to the best of our knowledge). Hence, our goal was to gain a complete insight into the complexity for several control and bribery problems and for various voting rules the more so as there are many other voting rules important in practice for which few complexity results are known

until now (such as Majority Judgment [8]). Thus, completing the "complexity puzzle" by including more and other voting rules seems to be a promising task.

In general, there are still many open problems in multimode control (adding, deleting, and bribing voters and/or candidates), such as multimode control in k-Approval ($k \geq 2$) and k-Veto ($k \in \mathbb{N}$). We only know that that (inexact) multimode voter control in Plurality is easy [81]. It therefore makes sense to solve these problems, especially in their most general versions (i.e., exact multimode control by adding + deleting voters). Likewise, one could combine bribery or other control problems (e.g., control by partition of voters) with control by adding/deleting candidates/voters. An appealing idea is studying multicontrol in a sense that a chair has several goals at the same time. So a chair might have the goal to see a candidate c win and another candidate d lose the election, he tries to reach that his three favorite candidates are among the top three scorers of the final election (when the underlying voting rule is a scoring rule). We refer again to [175] for a model where a chair tries to prevent several candidates from winning an election.

Aside from studying k-Approval and k-Veto, it would make sense to obtain dichotomy results for all scoring rules. According to this, there does not yet exist a dichotomy result for CCRV or exact voter control. Likewise, under the parallel-universe tie-breaking mechanism, one could examine which scoring rules combined with a runoff stage yield an easy bribery or voter control problem.

One could further study CCRV and other control problems under incomplete information—either in possible/necessary winner variants or directly applying a voting rule to partial votes. Especially for possible/necessary winner combined with CCRV/DCRV, it would be interesting to know whether it makes a difference if the registered or the unregistered voters are partial and the other group is complete. This would enable us to compare the results with the ones about control or bribery under partial information which we have achieved in [143, 70] and in Chapter 3, respectively. In this context, we point out that necessary constructive control by adding voters may be easy for the case where the registered votes are partial and the unregistered votes are complete, while the same problem is hard for the case where the unregistered votes are partial and the registered votes are complete. Possibly, such a result can be settled for replacement control as well.

Observe that one can investigate many other forms of strategic behavior in voting and show the missing results for various voting rules. Accordingly, a similar study to the one by us in [73] can be applied to control by partitioning voters or candidates, by runoff partition of candidates, by multipartition, equipartition, or partitioning groups (for other control models and the literature for these problems, we refer to Section 3.1). Likewise, various forms of bribery such as support bribery [150] or shift bribery [150] are worth further investigations. Possibly, new forms of control, bribery, or manipulation are discovered.

Finally, we again refer to other concepts in complexity theory, such as parameterized complexity or average-case complexity.

Part II

Group Identification

Chapter 6

Introduction

In this part, we present the results of our works about group identification [71, 72]. First of all, Section 6.1 provides the most important notions in group identification. Section 6.2 gives an overview of the literature in and around group identification.

6.1 Preliminaries

Let $N := \{a_1, \ldots, a_n\}$ be a set of $n \in \mathbb{N}$ *individuals*. For practical reasons, we mostly use a, a', or b to denote individuals in N in the following. A *profile* over N is a function $\varphi : N \times N \to \{0,1\}$. We say that individual $a \in N$ *qualifies (disqualifies)* $a' \in N$ if $\varphi(a,a') = 1$ ($\varphi(a,a') = 0$). The mapping φ induces a matrix $(\varphi) \in \{0,1\}^{n \times n}$ where $\varphi_{ij} := \varphi(a_i, a_j)$. A *social rule* is defined as a function $f : (\varphi, N) \to \mathscr{P}(N)$, i.e., it selects some individuals in N that are said to be *socially qualified* with respect to f and an instance (φ, N). Note that we can restrict f and φ to each subset $T \subseteq N$ by replacing N by T in the definition of a social rule. Next we define the social rules considered in Part II. We can roughly divide them into *consent rules* and *procedural rules*. The former class is specified by two parameters and is directly applied to an instance (φ, N). The latter rules iteratively amplify the set of socially qualified individuals and stop as soon as the set of socially qualified individuals does not change anymore. The main reason why we study consent rules, the liberal-start-respecting rule, and the consensus-start-respecting rule is that they are among the most significant social rules that have been investigated in the literature so far. Besides, they satisfy several fairness properties, see, e.g., [48, 111, 147]. The social rules studied by us are defined as follows (our definitions are in the style of the definitions in [171] the more so as our works can be regarded as a continuation of the work therein).

- **Consent Rule** ($f^{(s,t)}$). Consent rules are based on two parameters $s,t \in \mathbb{N}$ in a way that it holds for an individual set N and individual $a \in N$:

  - Individuals a qualifying themselves are socially qualified (formally, $a \in f^{(s,t)}(\varphi, N)$) if and only if $|\{a' \in N : \varphi(a',a) = 1\}| \geq s$ holds.
  - Individuals a disqualifying themselves are socially disqualified (formally, $a \notin f^{(s,t)}(\varphi, N)$) if and only if $|\{a' \in N : \varphi(a',a) = 0\}| \geq t$ holds.

The two parameters s and t are called *consent quotas*. For $s = t = 1$, we obtain the *liberal rule* f^L. According to the liberal rule, an individual is socially qualified if and only if he qualifies himself, i.e., we have $f^L(\varphi,N) = \{a \in N : \ \varphi(a,a) = 1\}$. The liberal rule is the only consent rule with the property that the question whether or not an individual a is socially qualified, only depends on a's self-evaluation and is independent from the others' valuations. Note that for our complexity analysis, unless stated otherwise, s and t are constant and do particularly not depend on the number of individuals. Nevertheless, we will check in several proofs whether the respective results also hold when some or all parameters become non-constant.

We point out that the original definition of consent rules by Samet and Schmeidler [147] contains the additional constraint $s+t \leq n+2$ for the consent quotas s and t which must hold as otherwise the *monotonicity property* is hurt. A social rule is *monotonic* if a socially qualified individual a is still socially qualified when someone who disqualifies a changes his valuation to qualify a and the remainder of the profile remains unchanged. In fact, this monotonicity condition can merely be hurt when an individual a changes his self-assessment from not qualified to qualified. In other words, even without the restriction $s+t \leq n+2$, if a is socially qualified, then he is still socially qualified if an individual other than a originally disqualifying a changes his assessment to qualify a. To illustrate the monotonicity property, consider the following example with individual set $N = \{a,b,c\}$, consent quotas $s = t = 3$, and a profile φ defined by $\varphi(a,a) = 0 = \varphi(b,a)$, $\varphi(c,a) = 1$ (the other values do not matter in this context). Observe that a is socially qualified under profile φ as a disqualifies himself and has at most two disqualifications in total. By setting $\varphi(a,a) = 1$ and leaving the remainder of the profile φ unchanged, there are two qualifications for a in total, but a is socially disqualified now, having a total of $2 < 3 = s$ qualifications.

As we examine several problems in Part II from the complexity theoretic point of view, we ignore this additional constraint and hence obtain even more general results.

- **Consensus-Start-Respecting Rule (f^{CSR}).** This rule is defined recursively by first determining a starting set of socially qualified individuals and then iteratively extending the set of socially qualified individuals until there is no change anymore. Formally, f^{CSR} is defined as follows. Let $K_0^C(\varphi,N)$ be the set of initially qualified individuals defined as:

$$K_0^C(\varphi,N) := \{a \in N : \ \forall a' \in N : \ \varphi(a',a) = 1\}.$$

Then we successively compute for natural numbers i:

$$K_i^C(\varphi,N) = \{a \in N : \ \exists a' \in K_{i-1}^C(\varphi,N) : \ \varphi(a',a) = 1\} \cup K_{i-1}^C(\varphi,N).$$

We obtain $f^{CSR}(\varphi,N) = K_i^C(\varphi,N)$ for some i with $K_i^C(\varphi,N) = K_{i+1}^C(\varphi,N)$.

- **Liberal-Start-Respecting Rule (f^{LSR}).** Again, we have a social rule iteratively defined as follows:

$$K_0^L(\varphi,N) := \{a \in N : \ \varphi(a,a) = 1\}.$$

The remaining iterations are computed as for f^{CSR}, i.e., via

$$K_i^L(\varphi,N) = \{a \in N : \ \exists a' \in K_{i-1}^L(\varphi,N) : \ \varphi(a',a) = 1\} \cup K_{i-1}^L(\varphi,N) \ (i \in \mathbb{N}).$$

Likewise, we obtain $f^{LSR}(\varphi,N) = K_i^L(\varphi,N)$ for some i with $K_i^L(\varphi,N) = K_{i+1}^L(\varphi,N)$.

Note that the starting set of f^{LSR} includes all individuals that qualify themselves, whereas the starting set of f^{CSR} contains only individuals qualified by each individual. In particular, we have $K_i^L(\varphi,N) \supseteq K_i^C(\varphi,N)$ for each $i \in \mathbb{N}_0$ and each instance (φ,N). When N and φ are clear from the context, we sometimes write K_0^L and K_0^C instead of $K_0^L(\varphi,N)$ and $K_0^C(\varphi,N)$, respectively.

Observe that for both procedural rules we can determine the socially qualified individuals in $O(n^2)$ time. The starting sets $K_0^C(\varphi,N)$ and $K_0^L(\varphi,N)$ can be computed in $O(n^2)$ and $O(n)$ time, respectively. For both rules, we have to check for at most n "pivotal individuals" a which individuals are qualified by a (which is possible in $O(n^2)$ in total). Moreover, searching for the pivotal rows of the matrix φ can be done in quadratic time as well. [1]

The following example visualizes the way how social rules work:

Example 6.1. *Let $N = \{a_1,a_2,a_3,a_4\}$. Consider the profile φ over N as follows (the entry row indexed by a_i and column indexed by a_j is $\varphi(a_i,a_j)$).*

	a_1	a_2	a_3	a_4
a_1	*1*	*1*	*1*	*1*
a_2	*0*	*1*	*1*	*0*
a_3	*0*	*1*	*0*	*0*
a_4	*0*	*1*	*1*	*0*

The socially qualified individuals with respect to some of the above social rules f are as follows.

$f^{(1,1)}$	$f^{(1,2)}$	$f^{(2,1)}$	$f^{(2,2)}$	f^{CSR}	f^{LSR}
a_1,a_2	a_1,a_2,a_3	a_2	a_2,a_3	a_2,a_3	a_1,a_2,a_3,a_4

Observe that the liberal rule $f^{(1,1)}$ selects exactly the individuals qualifying themselves, namely a_1 and a_2. For $f^{(1,2)}$, a_1 and a_2 are still socially qualified, qualifying themselves. Moreover, according to the definition, individuals a disqualifying themselves are only socially qualified if all other individuals qualify a. This is the case for a_3, but not for a_4. Thus, a_3 is socially qualified as well. For $f^{(2,1)}$, a self-disqualification of an individual immediately implies that this individual is socially disqualified. Individuals a qualifying themselves require at least two qualifications in total, consequently there must be at least one other individual that qualifies a. This holds for a_2, but not for a_1. The $f^{(2,2)}$ rule socially qualifies a_2 (at least two qualifications in total) and a_3 (at most one disqualification in total), the other two individuals do not satisfy their respective criteria.

For the f^{CSR} rule, we obtain the starting set $K_0^C = \{a_2\}$ (as only a_2 is qualified by everyone). As a_2 qualifies himself and a_3, we have $K_1^C = \{a_2,a_3\}$. Now we check which individuals, not yet socially qualified, are qualified by a_2 or a_3. There is no such individual and we therefore obtain $K_2^C = \{a_2,a_3\} = K_1^C$. Since there is no change anymore, it holds $K_2^C = f^{CSR}(\varphi,N) = \{a_2,a_3\}$ by definition. The f^{LSR} rule has a_1 and a_2 in its starting set. As a_1 qualifies all individuals, we have $K_1^L = \{a_1,a_2,a_3,a_4\} = N$. According to $K_2^L = N = K_1^L$, we obtain $f^{LSR}(\varphi,N) = N$ from the definition of the liberal-start respecting rule.

[1] Possibly, the complexity for determining the socially qualified individuals in procedural rules has already been existing, but we mention it just in case.

Observe that six different rules output five different subsets of socially qualified individuals. One can construct other examples with six different sets of socially qualified individuals. In the following example with $N = \{a_1, a_2, a_3, a_4\}$ and

$$\varphi = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix},$$

each pair of distinct social rules outputs different sets of socially qualified individuals. We obtain $f^{LSR}(\varphi, N) = N$, $f^{CSR}(\varphi, N) = \{a_1, a_3, a_4\}$, $f^{(1,1)}(\varphi, N) = \{a_1, a_2\}$, $f^{(1,2)}(\varphi, N) = \{a_1, a_2, a_4\}$, $f^{(2,1)}(\varphi, N) = \{a_1\}$, and $f^{(2,2)}(\varphi, N) = \{a_1, a_4\}$.

Moreover, it may occur that all six social rules from the previous examples yield one and the same set of socially qualified individuals. By way of example, the (4×4)-matrix with only zero entries canonically outputs the empty set as socially qualified individuals (except for degenerate examples with a consent rule and $t > n$).

Last but not least, we point out that all social rules can represent different scenarios. So the liberal rule applies to situations where the social status of an individual merely depends on his own opinion. Larger s and t reflect the majoritarian perspective for which the influence of the whole group is stronger than an individual's influence. E.g., in case an individual disqualifies himself, he can yet be socially qualified when the largest part of the group consider a as socially qualified. Compare the arguing in [147].

Accordingly, both procedural rules can be used to describe group structures. K_0^L and K_0^C can be regarded as the cores of a given group containing the individuals that regard themselves as qualified or that are considered as qualified by all individuals of the group. Especially individuals in the core of the consensus-start respecting rule may be regarded as the center of a group with the broadest support among all individuals. Following this, all individuals that are somehow directly or indirectly connected to individuals in the core have a certain status in N as well. Individuals occurring in sets K_i^C with smaller index could be regarded as closer to the core than other individuals. Nevertheless, one should take account of the total numbers of qualifications of the individuals as well. Since we investigate the complexity of several problems, such interpretations are immaterial to our analysis. In contrast to the f^{CSR} rule, all individuals regarding themselves as qualified belong to the core according to the liberal-start respecting rule—even when all other individuals do not qualify them. As suggested in [147], the f^{LSR} rule applies to situations where each individual's opinion about himself has a crucial impact. Note that other procedural rules are thinkable where every individual in the core satisfies a certain quota of the total number of qualifications from (other) individuals. Generally, especially procedural rules provide us some kind of network structure for the individual group N. Thus, bribery or control—studied in the next section—can be regarded as lobbying or influencing the "right" group members (for lobbying in voting, we refer to [25, 35]).

6.2 Related Work

Group identification itself, in particular the liberal rule, the consent rules, the consensus-start-respecting rule, and the liberal-start-respecting rule, was introduced and further studied from an economic and axiomatic point of view in [111, 47, 48, 130, 147]. In contrast, we investigate the

computational aspects of strategic behavior in group identification and group identification under partial information.

The model of group identification has also been studied in a recent paper in the context of expert selection in social networks [154]. However the work is different from ours the more so as the author proposes an algorithm based on some axioms and provides a case study, but does not consider the computational aspects.

Group identification is somewhat related to voting under approval-based voting rules in multi-winner elections as for two given individuals a and a', individual a either qualifies or disqualifies a'. In other words, all social rules studied by us are approval-based in a sense that each individual approves (qualifies) or disapproves (disqualifies) every individual. As opposed to approval voting [27], we do not count the number of approvals and let the individuals with the highest number of approvals win, but apply a social rule given the approval assignment. Another related work to mention is the work by Kilgour, Brams, and Sanver on electing representative committees via approval ballotting [114]. Again, they look for a fixed committee size. Furthermore, they do not consider the complexity of manipulative actions in their model.

Elkind et al. proposed and studied the properties of multi-winner voting rules based on single-winner scoring rules [53]. There are many different multi-winner voting rules based on single-winner voting rules in the literature that we do not use in our work [56, 84, 92, 142]. Meir et al. initiated the complexity theoretic analysis of strategic behavior in multi-winner elections where they investigated the complexity of manipulation and control under some prominent voting rules [126]. Obraztsova, Zick, and Elkind studied the complexity of manipulation in multi-winner scoring rules with a special focus on tie-breaking [135]. Furthermore, we point out the work by Aziz et al. on the computational aspects of best responses in multi-winner approval voting [4]. Last but not least, there are some recent works about multi-winner voting rules for which the size of the winner set is not predetermined [83, 113].

Our works fit in the line of research on the computational aspects of strategic behavior in elections initiated by a series of papers by Bartholdi et al. [14, 15, 16], proposing that computational hardness offers a (worst-case) protection against manipulative attacks. For the literature about bribery and electoral control, we refer to Section 3.1. In contrast to these works, we focus on bribery and control in group identification.

Constructive control by adding, deleting, and partitioning individuals for group identification were first introduced and studied by Yang and Dimitrov [171]. In Chapter 7, dealing with strategic behavior in group identification, we extend their work to destructive control and to constructive and destructive bribery. To the best of our knowledge, bribery has not been studied for group identification up to now.

Chapter 8 is about group identification with partial information. Our problems are somewhat related to the possible and necessary winner problems first introduced by [117]. As far as we know, the problem of determining the socially qualified individuals in group identification with partial information has not been investigated so far. Nevertheless, determining winners in voting rules with partial information has been studied in the literature. Once more, we refer to Section 3.1 presenting the related work in the context of voting under partial information.

There are various differences between voting and group identification. First of all, in group identification the sets of voters and candidates coincide and are called *individual set*. Each individual thus votes on every other individual and himself. Furthermore, in our setting individuals are neither

strategical nor selfish. Finally, group identification differs from single- or multi-winner voting as a social rule does not care about the size of the set of socially qualified individuals which can be arbitrary. In opposition, in multi-winner voting one often regards a committee of a fixed size. Thence, if at all, the social rules studied in this thesis are somewhat similar to single-winner approval-based voting rules that output all candidates as winners that satisfy a certain criterion. For example, regard voting rules based on a threshold where all candidates with an approval score above or not below this threshold are the winners. Such thresholds can be fixed numbers, but also be dependent on the number of voters. Possible thresholds are $|V|$ (each winner must be approved by all voters), $|V|/2$ (each winner is approved by at least half of all voters), or 0 (all candidates are winners). Once more, we point out the five- or ten-percent hurdles in many elections. Another example is given by the Weak Condorcet rule which does not care about the number of winners and any number of candidates can be in the set of winners. Nevertheless, the setting in group identification is different as candidates and voters are merged to the set of individuals and intuitively our focus lies on finding a group rather than in the context of voting where several co-winners are often considered as a necessary evil.

In contrast to voting rules, a social rule in the context of group identification does not aim at selecting winners, but provide quantitative criteria to decide if an individual is appropriate for a given task or not. Note that these criteria are fixed (e.g., for consent rules, two parameters uniquely determine these criteria) and can produce arbitrary numbers of socially qualified individuals, while in multi-winner elections we are interested in electing a committee of fixed size and in single-winner elections we canonically search for one winner (although in the co-winner model, a voting rule may output several winners as well).

Chapter 7

Strategic Influences in Group Identification

This chapter presents our results about strategic behavior in group identification [71]. We proceed as follows. In Section 7.1, we introduce the underlying problems. Section 7.2.1 is about constructive bribery, whereas Section 7.2.2 deals with destructive misuses on groups. Section 7.3 concludes the section.

7.1 Problem Settings

In this section, we introduce the formal definitions of the problems considered in this chapter. Constructive control by adding, deleting, and partitioning individuals in the context of group identification was introduced and studied in [171]. We extend their work by defining the destructive versions of these problems. In the destructive version of adding individuals, the chair's goal is to socially disqualify a distinguished set of individuals by adding some individuals to the group.

	f-DESTRUCTIVE GROUP CONTROL BY ADDING INDIVIDUALS (DGCAI)
Given:	A 5-tuple (N, φ, S, T, ℓ) of a set N of individuals, a profile φ over N, two nonempty subsets S and T such that $S \subseteq T \subseteq N$ and $S \cap f(\varphi, T) \neq \emptyset$, and a natural number ℓ.
Question:	Is there a subset $U \subseteq N \setminus T$ such that $\|U\| \leq \ell$ and $S \cap f(\varphi, T \cup U) = \emptyset$?

In the destructive version of deleting individuals, the chair removes some individuals in order to socially disqualify a distinguished set of individuals. Note that the chair is not allowed to remove individuals from the distinguished set.

	f-DESTRUCTIVE GROUP CONTROL BY DELETING INDIVIDUALS (DGCDI)
Given:	A 4-tuple (N, φ, S, ℓ) of a set N of individuals, a profile φ over N, a nonempty subset $S \subseteq N$ such that $S \cap f(\varphi, N) \neq \emptyset$, and a natural number ℓ.
Question:	Is there a subset $U \subseteq N \setminus S$ such that $\|U\| \leq \ell$ and $S \cap f(\varphi, N \setminus U) = \emptyset$?

Our final destructive control model is partitioning the set of individuals.

f-DESTRUCTIVE GROUP CONTROL BY PARTITION OF INDIVIDUALS (DGCPI)	
Given:	A 3-tuple (N,φ,S) of a set N of individuals, a profile φ over N, and a nonempty subset $S\subseteq N$ such that $S\cap f(\varphi,N)\neq\emptyset$.
Question:	Is there a subset $U\subseteq N$ such that $S\cap f(\varphi,V)=\emptyset$ with $V=f(\varphi,U)\cup f(\varphi,N\setminus U)$?

In contrast to control, in *bribery* an external agent is allowed to change some individuals' opinions over N in a way he desires.

f-CONSTRUCTIVE GROUP BRIBERY (CGB)	
Given:	A 4-tuple (N,φ,S,ℓ) of a social rule f, a set N of n individuals, a profile φ over N, a nonempty subset $S\subseteq N$ with $S\not\subseteq f(\varphi,N)$, and a natural number ℓ.
Question:	Is there a way to change at most ℓ rows of the matrix φ such that $S\subseteq f(\bar{\varphi},N)$ where $\bar{\varphi}\in\{0,1\}^{n\times n}$ is the resulting new profile?

By replacing $S\not\subseteq f(\varphi,N)$ by $S\cap f(\varphi,N)\neq\emptyset$ and $S\subseteq f(\bar{\varphi},N)$ by $S\cap f(\bar{\varphi},N)=\emptyset$, we obtain the definition of f-DESTRUCTIVE GROUP BRIBERY (f-DGB, for short) which asks whether the briber can change ℓ individuals' valuations in a way that all individuals in subgroup S are socially unqualified after the bribery.

We say that a social rule f is *immune* to a constructive (destructive) problem defined above if it is impossible to make every individual in S socially qualified (not socially qualified) by performing the corresponding manipulative action, i.e., the problem has only NO-instances. Otherwise, the social rule f is said to be *susceptible* to the problem.

Observe that our definitions exclude the possibility that in a constructive problem all individuals are socially qualified in the initial situation. Likewise, in destructive control or bribery at least one individual is socially qualified in the original profile.

We further point out that in all our problems, the parameter ℓ is strictly positive (except for the partitioning cases where ℓ does not occur). The complexity results in the next sections also hold when ℓ is a nonnegative integer. In case $\ell=0$, however, our problems would be trivial and reduce to evaluating a social rule. Hence, we exclude the cases where $\ell=0$.

Last but not least, we have to mention that a social rule f is not part of the input. Nevertheless, various complexity results in this and the next chapter still hold when s or t are not constant (i.e., the consent rule $f^{(s,t)}$ may belong to the problem's input and the complexity remains polynomial).

7.2 Results

7.2.1 Constructive Group Bribery

Throughout this section, we let (N,φ,S,ℓ) denote a bribery instance where N is a set of n individuals ($n\in\mathbb{N}$), φ a profile, $S\subseteq N$ a non-empty set of distinguished individuals the briber wants to make socially qualified, and ℓ is a natural number representing the maximum number of individuals the briber may bribe. $\bar{\varphi}$ denotes the profile after the bribery. Moreover, in some proofs throughout this chapter, we make use of the sets $S_h:=\{a\in S:\ \varphi(a,a)=h\}$ ($h=0,1$).

Our results on constructive group bribery are summarized in Table 7.1. The first result indicates that bribery in consent rules is easy given $t=1$.

f	$f^{(s,1)}$	$f^{(s,2)}$	$f^{(s,t)}$, where $s \geq 1$ and $t \geq 3$	f^{CSR}	f^{LSR}
f-CGB	P		NPC	P	P

Table 7.1: Results for constructive group bribery. "P" stands for "polynomial-time solvable" and "NPC" for "NP-complete".

Theorem 7.1. $f^{(s,1)}$-CGB *is in* P *for all constants s.*

Proof. We first compute $S_0 = \{a \in S\colon \varphi(a,a) = 0\}$ and $S_1 = S \setminus S_0$. These computations take $O(n)$ time since our algorithm simply runs all diagonal elements of the matrix φ according to individuals in S. Our algorithm then checks the following cases:

- $\ell < |S_0|$. Then our instance is a NO instance as there is an $a \in S$ disqualifying himself after the bribery and consequently $a \notin f(\bar{\varphi}, N)$ after the bribery. (Observe that this holds regardless of the final profile $\bar{\varphi}$.)
- $\ell \geq |S_0|$. In this case, the briber bribes all individuals in S_0 and makes all of them qualify all individuals (at least they all must qualify all individuals in S). For $s = 1$, our instance is a YES instance because each individual in S qualifies himself and this is sufficient for S to be socially qualified. For $s > 1$, we must ensure that each individual qualifying himself has s qualifications in total. As there are still $\ell - |S_0|$ bribes left and $S = S_1$ currently holds (i.e., each individual in S qualifies himself), we may w.l.o.g. restrict ourselves to a (transformed) bribery problem with $S_0 = \emptyset$ (where all already bribed individuals qualify all individuals). There are still two cases left to differentiate between:
  - $\ell \geq s$. Then the briber bribes arbitrary s individuals and makes them all qualify all individuals. Thus—due to our assumption $\varphi(a,a) = 1$ for each $a \in S$—it holds $S \subseteq f(\bar{\varphi}, N)$ after the bribery as each $a \in S$ has s or more qualifications in total.
  - $\ell < s$. Now ℓ is bounded from above by the constant s. It suffices to check for all $\binom{n}{\ell}$ ways to bribe ℓ individuals (there are $O(n^{s-1})$ such possibilities to check) if all individuals in S are socially qualified provided that the briber makes each bribed individual qualify all individuals (in S).

  For $s = 1$, the complexity is therefore linear in n. Given $s > 1$, it first takes us $O(n)$ time to compute S_1 and S_0 and then $O(n^2 \cdot n^{s-1}) = O(n^{s+1})$ time to check for each combination of $\ell < s$ bribes whether all individuals in S are socially qualified after the bribery or not.

❑

In contrast, we cannot preserve easiness by fixing $s = 1$ and permitting t to be an arbitrary nonnegative integer. We show hardness via reduction from RX3C.

Theorem 7.2. $f^{(s,t)}$-CGB *is* NP*-complete for each* $s \geq 1$ *and* $t \geq 3$.

Proof. We prove our theorem for $s = 1$ and $t = 3$ and will argue later how to adjust the proof for higher values s and t. Given an RX3C instance by a pair $(B, \mathscr{S})$ defined as in the RX3C definition,

we construct the following instances of $f^{(1,3)}$-CGB with distinguished subgroup $S = B$, bribery limit $\ell = n - m = 2m$ (due to $n = 3m$), and individuals $N = B \cup \mathscr{S}$. The profile φ is defined as follows:

- Each $b_j \in B$ disqualifies himself and qualifies all other individuals.
- S_i disqualifies b_j if and only if $b_j \in S_i$.
- The values $\varphi(S_i, S_h)$ may be arbitrary ($1 \leq i, h \leq n$, $i \neq h$).

Each individual in B has four disqualifications in total. In order to be socially qualified, an individual in B must either qualify himself or have at most two disqualifications after the bribery, i.e., at least two disqualifications must fall away by the bribery. Notice that a bribed individual qualifies all individuals including himself (at least the individuals in S). We claim that there is an exact cover of B if and only if there exists a successful bribery.

($\Rightarrow$): Assume that $\mathscr{S}' \subseteq \mathscr{S}$ is an exact cover of B. Then the briber bribes all individuals in $\mathscr{S} \setminus \mathscr{S}'$ such that they qualify all individuals. Thus one disqualification from individuals in $\mathscr{S}$ remains for each b_j. Consequently—since each b_j disqualifies himself—b_j has two disqualifications in the end, and so $S \subseteq f^{(1,3)}(\bar{\varphi}, N)$.

($\Leftarrow$): Now assume that there is a successful bribery. First we show that the briber does not bribe any individuals in B. Second, we prove that a successful bribery in $\mathscr{S}$ requires the existence of an exact cover of B.

Suppose that the briber bribes α individuals in B and $2m - \alpha$ individuals in $\mathscr{S}$. Notice that, as mentioned above, each bribed individual qualifies all individuals after the bribery. The b_j bribed are socially qualified (as they qualify themselves after the bribery) and can be ignored for the remaining bribes. To successfully bribe, the briber must take away at least two disqualifications from each remaining $b_j \in B$, i.e., $2(3m - \alpha)$ disqualifications in total. As there are only $2m - \alpha$ bribes left and three disqualifications are caught by each bribe (i.e., $3(2m - \alpha)$ in total), a necessary condition for a successful bribery is that $2(3m - \alpha) \leq 3(2m - \alpha)$, this implies that $\alpha \leq 0$ or—as α is a nonnegative integer—$\alpha = 0$, i.e., we can deduce that the briber bribes only individuals in $\mathscr{S}$.

As he must take away at least two disqualifications from all $3m$ individuals in B, and $2m$ bribes take away $2m \cdot 3 = 6m$ disqualifications in total, this implies that each $b_j \in B$ loses exactly two disqualifications and this in turn means that exactly one disqualification remains in the unbribed individuals in $\mathscr{S}$ for each $b_j \in B$. Thus the individuals in $\mathscr{S}$ not bribed form an exact cover of B.

For $t \geq 4$, we set $\ell = 2m + (t - 3)$ and add $t - 3$ dummy individuals in S disqualifying each individual and being disqualified by every individual. These dummy individuals are bribed with the highest priority. The remainder of the proof remains identical.

For $s \geq 2$, the same reasoning as for $s = 1$ can be used: s as a parameter is only relevant when $b_j \in B$ is qualified after the bribery. As bribing b_j and making him qualify himself is too costly given our construction (i.e., bribing only one $b_j \in B$ makes it impossible that all other individuals in B lose their disqualifications), for $s \geq 2$ our focus lies on making each individual in B fall below the threshold t, too. ❑

In contrast, constructive group bribery is easy for procedural rules.

Theorem 7.3. *Both f^{LSR}-*CGB *and f^{CSR}-*CGB *are solvable in polynomial time.*

Proof. For f^{LSR}, our instance is always a YES instance: the briber simply bribes an arbitrary individual $\hat{a}$ and makes $\hat{a}$ qualify himself and all individuals in S. This ensures that $\hat{a}$ is in the starting set and each individual $a \in S$ is socially qualified as a is qualified by $\hat{a}$, i.e., we have $K_0^L(\bar{\varphi},N) \supseteq \{\hat{a}\}$, $K_1^L(\bar{\varphi},N) \supseteq \{\hat{a}\} \cup S$. Actually, when $\hat{a}$ qualifies all individuals (including the individuals in $N \setminus S$), we even have $K_1^L(\bar{\varphi},N) = N$, with only one individual being bribed.

Given f^{CSR}, we first check if $K_0^C \neq \emptyset$ (which requires $O(n^2)$ time). If yes, we bribe any $a \in K_0^C$ and make a qualify all individuals including himself. This makes all individuals (and in particular the individuals in S) socially qualified. If no, we distinguish the following cases:

- Let $y(a)$ be the number of individuals a' with $\varphi(a',a) = 1$. The values $y(a)$ can be computed in quadratic time in n. If $\max_{a \in N} y(a) + \ell < n$, there is no way to make any $a \in S$ socially qualified as even the individual with the highest number of qualifications in the original election cannot reach K_0^C after the bribery (even if all bribed individuals disqualify this individual before and qualify him after the bribery). As $K_0^C = \emptyset$ after the bribery, there is no chance for the briber to reach his goal since due to the definition of the f^{CSR} rule, $K_0^C = K_i^C = \emptyset$ for all natural numbers i holds.

- For $\max_{a \in N} y(a) + \ell > n$, the briber chooses an individual $\hat{a}$ with $y(\hat{a}) = \max_{a \in N} y(a)$, bribes $\hat{a}$ and all remaining individuals initially not qualifying $\hat{a}$, and makes each bribed individual qualify w.l.o.g. all individuals in N. This makes all individuals in S socially qualified.

- If $\max_{a \in N} y(a) + \ell = n$ (i.e., an individual with the maximum number of qualifications can barely reach the starting set K_0^C after the bribery), we check for each $\hat{a}$ maximizing this expression (there may be more than one such individual) if making $\hat{a}$ reach K_0^C implies that all individuals in S belong to some K_i^C ($i \in \mathbb{N}_0$). The briber bribes all individuals initially disqualifying $\hat{a}$ and makes each bribed individual qualify all individuals. As the individuals to bribe are fixed, our problem reduces to evaluating the f^{CSR} rule. This subcase requires at most $O(n^3)$ time: The briber bribes the remaining individuals initially not qualifying a (which is possible in quadratic time) by filling the corresponding rows of the matrix φ with ones, then we check by computing the socially qualified individuals for f^{CSR} if all individuals in S are socially qualified (observe that f^{CSR} can be evaluated in quadratic time in n).

❑

Note that the complexity of $f^{(s,2)}$-CGB is still open. One can verify that the problem is easy unless $s < \ell < |S_0|$, i.e., to show hardness one needed to regard this open subcase. The remaining cases are easy to decide since for $s \geq \ell$, the number of ways the briber can bribe ℓ individuals in bounded from above by a polynomial since s is a constant. If $|S_0| \leq \ell$ and $\ell > s$, we accept as the briber can ensure that all individuals in S qualify themselves after the bribery and ℓ bribes guarantee that all individuals in S have s or more qualifications in total.

7.2.2 Destructive Group Actions

In this section, we consider destructive influences on group identification, i.e., the briber or the chair seeks to prevent all individuals in a certain group $S \subseteq N$ from being socially qualified. Our results are summarized in Table 7.2. The first result gives a connection between the constructive

f	$f^{(s,t)}$							f^{LSR}	f^{CSR}
	$s=1$		$s=2$			$s\geq 3$			
	$t=1$	$t\geq 2$	$t=1$	$t=2$	$t\geq 3$	$t=1$	$t\geq 2$		
f- DGCAI	I	NPC	I	NPC	NPC	I	NPC	**I**	**NPC**
f- DGCDI	I	I	P	P	P	NPC	NPC	**P**	**P**
f- DGCPI	I	I	NPC	NPC	NPC	NPC	NPC	**P**	
f- DGB	P	P				NPC	NPC	**P**	**P**

Table 7.2: Results for destructive group control and destructive group bribery. Results for destructive group control for consent rules follow from Theorem 7.4 and Theorem 7.5, and the results in [171]. Results for destructive group bribery for consent rules follow from Theorem 7.4, Theorem 7.5, and Table 7.1. Results for procedural rules are written in boldface and are shown in Theorem 7.6, 7.7, 7.8, 7.9, and 7.11 in this section as they cannot be derived from Theorem 7.4 and 7.5. In the table, "I" stands for "immune", "P" for "polynomial-time solvable", and "NPC" for "NP-complete".

and destructive variants of our problems for consent rules. We point out that Theorem 7.4 and Theorem 7.5 only hold for constant s and t.

Theorem 7.4. *[147] Let φ be a profile over N. Then it holds $f^{(s,t)}(\varphi,N) = N \setminus f^{(t,s)}(-\varphi,N)$ for a given consent rule $f^{(s,t)}$, where $-\varphi$ is obtained from φ by reversing the values (i.e., $\varphi(a,b) = 1$ if and only if $-\varphi(a,b) = 0$).* [1]

Proof. Given an instance (φ,N), let $a \in f^{(s,t)}(\varphi,N)$. In case $\varphi(a,a) = 1$, then there are at least s individuals qualifying a with respect to φ. Hence, there are at least s individuals disqualifying a with respect to $-\varphi$. As $-\varphi(a,a) = 0$, $a \notin f^{(t,s)}(-\varphi,N)$. Given $\varphi(a,a) = 0$, then at most $t-1$ individuals in N disqualify a (otherwise a is socially disqualified for φ). It follows that $-\varphi(a,a) = 1$ and at most t individuals in N qualify a in the profile $-\varphi$. Hence, a is not socially qualified in the profile $-\varphi$ according to the social rule $f^{(t,s)}$.

It follows by means of a similar argument that the condition $a \notin f^{(t,s)}(-\varphi,N)$ implicates that $a \in f^{(s,t)}(\varphi,N)$. ❑

As a direct consequence of Theorem 7.4, we can reduce constructive group bribery/control problems to destructive group bribery/control problems. As an example, if the chair tries to make an individual a socially qualified with respect to the $f^{(10,4)}$ rule and a profile φ by adding some individuals, he faces exactly the same problem as for the case where he wants to prevent a from being socially qualified for the $f^{(4,10)}$ rule and reversed φ function. Let AI, DI, and PI denote adding individuals, deleting individuals, and partitioning of individuals, respectively. In general, the following link between destructive and constructive group control/bribery holds.

Theorem 7.5. *Suppose that $X \in \{AI, DI, PI\}$. Then* $\begin{pmatrix} f^{(s,t)}\text{-CGCX} \\ f^{(s,t)}\text{-CGB} \end{pmatrix}$ *is* $\begin{pmatrix} \textit{in P} \\ \text{NP-}\textit{complete} \\ \textit{impossible} \end{pmatrix}$ *if and*

[1] This result also follows from Proposition 2 in [147], albeit in a different context.

only if $\begin{pmatrix} f^{(t,s)}\text{-DGCX} \\ f^{(t,s)}\text{-DGB} \end{pmatrix}$ *is* $\begin{pmatrix} \textit{in P} \\ \text{NP-}\textit{complete} \\ \textit{impossible} \end{pmatrix}$.

For the corresponding results, we refer to Table 7.2. Note that for $s = t$, the complexities for the constructive and destructive variants coincide. Theorem 7.4 and Theorem 7.5 do not apply to f^{CSR} and f^{LSR} as we can see in the following.

In the remainder of this section, we first examine destructive group control before we regard destructive group bribery. We restrict ourselves to procedural rules as the corresponding results for consent rules follow from Theorem 7.4 and 7.5.

Our first control result shows that adding individuals in the destructive case for the consensus-start-respecting rule is computationally hard. We prove the theorem via reduction from EXACT COVER BY 3-SETS (X3C).

Theorem 7.6. f^{CSR}-DGCAI *is* NP-*complete.*

Proof. Given an X3C instance $(B, \mathscr{S})$, we construct an instance (N, φ, S, T, ℓ) of f^{CSR}-DGCAI with $N = \mathscr{S} \cup B$, $T = S = B$, $\ell = m$, and φ defined as follows:

- For all $r, j \in \{1, \ldots, 3m\}$, let $\varphi(b_r, b_j) = 1$.
- For all $i, h \in \{1, \ldots, n\}$, let $\varphi(S_i, S_h) = 0$.
- If $b_j \in S_i$, for all $1 \leq i \leq n$ and $1 \leq j \leq 3m$, let $\varphi(S_i, b_j) = 0$; otherwise, let $\varphi(S_i, b_j) = 1$.

The values $\varphi(b_j, S_i)$ may be arbitrary. Note that each $b_j \in B$ is qualified by each individual in B. Thus all individuals in B are socially qualified at the beginning. We claim that an exact cover of B exists if and only if there is a successful control.

($\Rightarrow$): Suppose that there is an exact cover for B, i.e., there is a subset $\mathscr{S}' \subseteq \mathscr{S}$ so that each $b_j \in B$ is in precisely one $S_i \in \mathscr{S}'$. By adding the individuals in $\mathscr{S}'$, we achieve $K_0^C = \emptyset$ and thus $f^{CSR}(\varphi, B \cup \mathscr{S}') = \emptyset$. Each added S_i disqualifies himself and is thus not qualified by each individual. The same holds for each $b_j \in B$ as we have added an S_i according to the exact cover with $\varphi(S_i, b_j) = 0$.

($\Leftarrow$): Suppose that each individual $b_j \in B$ can be prevented from being socially qualified. As all individuals in B qualify themselves and $K_0^C = B$ holds in the original situation, for each $b_j \in B$, we must add a subset S_i with $\varphi(S_i, b_j) = 0$ to throw b_j out of the starting set. As $3m$ individuals require at least one disqualification from added individuals and m added individuals assign a total of $3m$ disqualifications, this in turn implies that all $b_j \in B$ are disqualified by exactly one S_i added. Consequently, the added S_i form an exact cover of B. ❑

The following theorem says that group control by adding individuals in f^{LSR} is never possible.

Theorem 7.7. f^{LSR} *is immune to destructive group control by adding individuals.*

Proof. Let (N, φ, S, T, ℓ) be an instance of f^{LSR}-DGCAI. First suppose that there is an $a \in S$ with $\varphi(a, a) = 1$. Then $a \in K_i^L$, for each nonnegative integer i, and this cannot be changed by adding additional individuals. Thus the only chance for the chair to reach his goal at all is when $\varphi(a, a) = 0$ holds for all $a \in S$ at the beginning. $a \in f^{LSR}(\varphi, N)$ implies that there are individuals $a_{i_0}, a_{i_1}, \ldots, a_{i_r}$

where r is a nonnegative integer, $a_{i_0} \in K_0^L$, $a_{i_j} \in N$ for $0 \leq j \leq r$, $\varphi(a_{i_j}, a_{i_{j+1}}) = 1$ for $0 \leq j \leq r-1$, and $\varphi(a_{i_r}, a) = 1$, i.e., there is a sequence of individuals qualifying the respective successors with starting point in K_0^L and endpoint in a. Such sequences exist for all socially qualified individuals $a \in S$ (there is at least one socially qualified individual in S according to the definition of DGCAI) and do not change no matter which additional individuals are added. Thus, we have immunity for this case. ❑

In contrast, destructive group control by deleting individuals in both f^{CSR} and f^{LSR} is possible and solvable in polynomial time. In order to show membership in P, we give reductions from our problems to the MINIMUM (u,u')-SEPARATOR problem which is known to be in P [152].

Theorem 7.8. *Both* f^{LSR}-DGCDI *and* f^{CSR}-DGCDI *are in* P.

Proof. Let (N, φ, S, ℓ) be a given instance of f^{LSR}-DGCDI. We construct the following auxiliary digraph G. An individual $a \in N$ yields vertex $v(a)$. There is an edge from vertex $v(a)$ to vertex $v(a')$, $a, a' \in N$, $a \neq a'$, if and only if a qualifies a'. Let $\mathcal{V}^{LSR}$ be the set of all vertices $v(a)$ with $\varphi(a,a) = 1$ and let $\Gamma_G(\mathcal{V}^{LSR})$ be the set of all vertices reachable from a vertex in $\mathcal{V}^{LSR}$. Then the socially qualified individuals with respect to f^{LSR} are the ones corresponding to vertices in $\Gamma_G(\mathcal{V}^{LSR})$. Due to this observation, we can solve the instance as follows. If $v(a) \in \mathcal{V}^{LSR}$ for some $a \in S$, the given instance is a NO instance. Otherwise, merge all vertices corresponding to individuals in S into $v(S)$, create a new vertex w and an edge from w to each vertex in $\mathcal{V}^{LSR}$. Let $\mathcal{V}'$ be a minimum $(w, v(S))$-separator. If $|\mathcal{V}'| \leq \ell$, the given instance is a YES instance; otherwise, it is a NO instance. Observe that this problem is in P as the computation of the help graph and all decisions require $O(n^2)$ time. Moreover, we know from [152] that computing a minimum separator can be done in polynomial time.

In the case of consensus-start-respecting rules, let (N, φ, S, ℓ) be a given instance of f^{CSR}-DGCDI. Again, we create an auxiliary digraph G identical to the graph in the first part of the proof, with the following difference. Instead of $\mathcal{V}^{LSR}$, we define $\mathcal{V}^{CSR}$ as the set of all vertices $v(a)$ for which the corresponding individual a is in the starting set, i.e., for which $\varphi(a', a) = 1$ holds for all individuals $a' \in N$. Then, the socially qualified individuals with respect to f^{CSR} are the ones corresponding to vertices in $\Gamma_G(\mathcal{V}^{CSR})$. Based on this observation, we solve the instance as follows. If there is an $a \in S$ such that $v(a) \in \mathcal{V}^{CSR}$, our instance is a NO instance. Otherwise, we merge all vertices corresponding to individuals in S into $v(S)$, create a new vertex w and an edge from w to every vertex in $\mathcal{V}^{CSR}$. Let $\mathcal{V}'$ be a minimum $(w, v(S))$-separator. If $|\mathcal{V}'| > \ell$, the given instance is a NO instance. If $|\mathcal{V}'| \leq \ell$ and $|\mathcal{V}^{CSR}| > |\mathcal{V}'|$, our instance is a YES instance. In fact, deleting the individuals according to vertices in $\mathcal{V}'$ makes all $a \in S$ socially disqualified. The reason is as follows. First, $\mathcal{V}^{CSR} \setminus \mathcal{V}' \neq \emptyset$. Assume for the sake of contradiction that after deleting all individuals standing for vertices in $\mathcal{V}'$ some $a \in S$ is still socially qualified. Due to the definition of the graph, the vertex $v(S)$ is still reachable from some vertices in a set U whose corresponding individuals are in the starting set of socially qualified individuals after deleting individuals corresponding to vertices in $\mathcal{V}'$. Moreover, all individuals in $\mathcal{V}^{CSR} \setminus \mathcal{V}'$ qualify individuals corresponding to U. Thus, there are edges from $\mathcal{V}^{CSR} \setminus \mathcal{V}'$ to U, and hence there are edges from w to $v(S)$ which is a contradiction. On the other hand, if $|\mathcal{V}'| \leq \ell$ but $|\mathcal{V}^{CSR}| \leq |\mathcal{V}'|$, we distinguish two cases. If there is a minimum $(w, v(S))$-separator $\mathcal{V}'$ such that $\mathcal{V}^{CSR} \setminus \mathcal{V}' \neq \emptyset$, the given instance is a YES instance as the individuals corresponding to $\mathcal{V}'$ is a solution (we may argue analogously to the case where $|\mathcal{V}'| \leq \ell$

and $|\mathcal{V}^{CSR}| > |\mathcal{V}'|$). If $\mathcal{V}^{CSR}$ is the unique minimum $(w, v(S))$-separator, we delete all individuals according to vertices in $\mathcal{V}^{CSR}$, reset $\ell := \ell - |\mathcal{V}^{CSR}|$ and repeat the algorithm with the new instance after the deletion of the individuals. ❑

We now turn to the partition of individuals cases.

Theorem 7.9. f^{LSR}-DGCPI *is in* P.

Proof. We let (N, φ, S) be a f^{LSR}-DGCPI instance. First, in case there is an $a \in S$ with $\varphi(a,a) = 1$, our instance is a NO instance: $a \in \bigcup_{i \in \mathbb{N}_0} K_i^L$ and this holds for each subset of N containing a. Thus, a survives—qualifying himself—both the preliminary and the final round and is therefore socially qualified being in the starting set independent from the other individuals' decisions. Thence, the only case worth studying is $\varphi(a,a) = 0$ for all $a \in S$. In this case, our instance is a YES instance: The partition (U,W) of N with $U = S$ and $W = N \setminus S$ ensures that neither in S survives the preliminary round (due to $K_0^L(\varphi, S) = \emptyset$) and thus no individual from S is socially qualified. As our problem merely reduces to the question whether there exists an $a \in S$ such that $\varphi(a,a) = 1$ (which can be checked in linear time in n), it follows that our problem is in P. ❑

The problem f^{CSR}-DGCPI is still open, however, we can show easiness under the restriction $\varphi = \varphi^T$ (i.e., a qualifies a' if and only if a' qualifies a).

Theorem 7.10. f^{CSR}-DGCPI *is in* P *if* $\varphi = \varphi^T$ *holds.*

Proof. We are given an instance (N, φ, S) with $\varphi = \varphi^T$. Observe that the starting set K_0^C is non-empty due to the definition of DGCPI, premising that at least one individual in S is socially qualified according to the profile φ. This in turn is impossible when the starting set is empty. Our algorithm checks the following cases:

- $K_0^C \cap S \neq \emptyset$. Then $a \in K_0^C \cap S$ is in the starting set for each subset of N, reaches the final round and is hence socially qualified. Thus, we reject in this case.
- $K_0^C \cap S = \emptyset$, $K_0^C \neq \emptyset$. Accordingly, there is some $a_0 \in K_0^C \setminus S$, i.e., $a_0 \notin S$ and a_0 is qualified by each individual in N. This implies that a_0 is in the starting set for each subgroup $N' \subseteq N$. Due to $\varphi = \varphi^T$, we have $\varphi(a_0, a) = 1$ for each $a \in N$ and especially each $a \in S$. Notice that each $a_0 \in K_0^C$ must be in a partition set different from each individual in S. Otherwise, a_0—socially qualified in his partition set—ties all S individuals belonging to the same partition set to the final round which makes them socially qualified, too (as a_0 is in the starting set in the final round, too). Consequently, we w.l.o.g. set $S \subseteq U$ and $K_0^C \subseteq W$. Now we check if there is some $a \in S$ qualified by each $a' \in S$. If no, we may set $U = S$ and $W = N \setminus S$ and—as no S individual survives the prelim—are done and accept. If yes, we do the following. Check if there is currently some $a \in N \setminus (U \cup W)$ being qualified by each individual in $N \setminus W$. Such individuals are qualified by each subgroup of $N \setminus W$ (and thus by each individual in U including all $a \in S$), reach the final round and—as φ is symmetric—each other element in U including S, too. Thus, the chair must successively set these individuals into W. In each iteration, we therefore check if there is some individual not yet assigned to U or W that is qualified by each individual not in the present set W. If yes, we fix this element in W and proceed with the next iteration. If no, we are done and put all still unassigned individuals

(if any) in U. For these individuals, we know that they are not qualified by all individuals in $N \setminus W = U$. Thus, they all do not belong to the starting set in U. If there is a chance at all to get each $a \in S$ out of the starting set in U (and make all individuals in S socially disqualified according to this), this procedure is the best strategy for the chair as he adds as many individuals from $N \setminus S$ as possible to the partition set U (i.e., the chance that for each $a \in S$ an individual disqualifying a belongs to the same partition set is thus maximized). After putting each individual in N either into U or W, it hence suffices to check if the starting set in U is empty or not (note that at most individuals in S can belong to the starting set in U after applying this algorithm). If yes, our instance is a YES instance. If no, we obtain a NO instance as all individuals but S are set into W, and hence some $a \in S$ and (due to the symmetry of φ) all individuals in S reach the final round and are socially qualified (as some $a_0 \in K_0^C$ survives the partition set W, reaches the final round and in turn makes all individuals in S socially qualified in the final round). Note that this algorithm requires $O(n^3)$ time as we regard $O(n)$ pivotal individuals in $N \setminus (K_0^C \cup S)$ one by one and verify in each step if all remaining $O(n)$ individuals not yet regarded are qualified by all $O(n)$ individuals in $N \setminus W$.

Observe that the problem (especially the algorithm in the last subcase) takes no more than $O(n^3)$ time as the first case (mainly computing the starting set) is in $O(n^2)$ and the algorithm in the second case works in cubic time. ❑

The assumption $\varphi = \varphi^T$ makes sense in practice and induces several interesting substructures. Individuals may build coalitions and hence there is mutual support (or rejection). Another, possibly more restrictive interpretation of a symmetric φ is that similar individuals (or individuals within a certain distance threshold) qualify each other.

Our last result discloses that destructive group bribery is easy for both procedural rules.

Theorem 7.11. f^{CSR}-DGB *and* f^{LSR}-DGB *are in* P.

Proof. Destructive group bribery in f^{CSR} is always possible (since we assume in our bribery definition that the briber can bribe at least one individual). The briber simply bribes an arbitrary individual and makes him disqualify each individual. By this, we obtain $K_0^C = \emptyset$.

For f^{LSR}, our input is an instance (N, φ, S, ℓ). A necessary condition for the briber to reach his goal is obtaining $S_1 = \emptyset$ after the bribery. We may assume that each bribed individual disqualifies all individuals after the bribery in order to keep the number of qualifications as low as possible in the final election. Our algorithm checks the following cases:

- $|S_1| > \ell$. Then our instance is a NO instance as the briber cannot reach $K_0^L \cap S = \emptyset$ after the bribery.
- $|S_1| = \ell$. Then the briber bribes all S_1 individuals, each bribed individual disqualifies all individuals after the bribery, and it remains to validate if some $a \in S$ is socially qualified or not in the final election.
- $|S_1| < \ell$. The briber then bribes all individuals in S_1 (these bribes are fixed) and $\ell - |S_1|$ further individuals. Thus, w.l.o.g., we regard an instance with $S_0 = S$ (and all individuals in S_1 in the initial situation are meant to disqualify all individuals in the transformed instance). We define the same help graph to the one in proof of Theorem 7.8.

It remains to argue that there is a minimum $(w, v(S))$-separator of size less or equal to ℓ if and only if there is a successful bribery of at most ℓ individuals. Bribing an individual and making him disqualify each individual can be interpreted as deleting this individual: let $a \in N \setminus (K_0^L \cup S)$, then, after the bribery, all edges outgoing from $v(a)$ are deleted (as the briber makes a disqualify all individuals after the bribery). Although there may be some ingoing edges left, they are worthless as $v(a)$ is a dead end and no other vertex is reached by $v(a)$. Thus, a can be treated as if deleted. A reasonable briber would not bribe individuals in S in doubt as his aim is to destroy all ingoing edges with higher priority than edges between some individual in S and other individuals (possibly in S). The reason is that it does not matter if one or more individuals in S are reached, i.e., we want to cut off the "first" connection to individuals in S.

Verify that the algorithm takes polynomial time as all computations and decisions except the last subcase require $O(n^2)$ time and computing a minimum separator is in P, too. ❑

Due to Theorem 7.4 and Theorem 7.5, we directly obtain the complexities of destructive group bribery in consent rules. We refer the reader to Table 7.2

7.3 Conclusion

We have investigated the complexity of destructive group control and constructive and destructive group bribery in the context of group identification. Constructive group control had been previously considered in the work by Yang and Dimitrov [171]. We have found a direct connection between destructive and constructive problems for consent rules and could thus derive all complexities for the destructive problems of consent rules. According to our results and the results obtained in [171], every consent rule $f^{(s,t)}$ with $s,t \geq 3$ resists all group control and bribery problems studied so far in the literature. In contrast, many group control and bribery problems for the liberal-start-respecting rule and consensus-start-respecting rule turned out to be polynomial-time solvable. Hence, our results suggest that consent rules outperform the liberal-start-respecting and consensus-start-respecting rules in terms of the resistance to strategic behavior.

We have not considered *manipulation* [14, 170] (i.e., referring to the setting in group identification, some strategic individuals report insincere preferences in order to make a desired group of individuals socially qualified). Given constructive group manipulation, the best a manipulator can do is qualifying all individuals. Likewise, the manipulators disqualify all individuals in the destructive version. These problems all are in P.

For future research, we refer to the open problems in this chapter. As an example, we mention constructive group bribery in $f^{(s,2)}$ respectively the "mirrored" problem in destructive group bribery. One could further extend the model to other social rules or other strategical influences on groups such as runoff partition of individuals or other control models [65].

Another direction for future research would be to consider some variants of the group bribery problems studied in this chapter. For instance, the briber has to pay 1 dollar for an individual to change his opinion over an individual and the briber wants to pay in total ℓ dollars in order to reach his goal. In addition, it is natural to assume in some setting that all individuals do not want to deviate much from their true opinions. In this setting, a bribed individual only flips at most τ of his

qualifications or disqualifications over the individuals, where τ is a small number. It also seems to be worth studying what happens when each individual qualifies or disqualifies exactly k individuals or when each individual has more than two options beyond qualification and disqualification. At this point, we have to mention the work by Cho and Ju [34] who studied *multinary group identification* where each individual a has an opinion about each individual b about the group to which b belongs (there may be more than two groups; for two groups, their model coincides with the classical model of group identification). One could also consider social rules in settings where the individuals provide further information, e.g., declare complete or partial rankings over the individuals in N.

Another intriguing task for future research is investigating bribery and control in group identification under settings with some uncertainty in the individuals' valuations. Possibly, some individuals are not able or not willing to express an opinion about all individuals or an external, manipulative agent has no access to all individuals' opinions. One such model of partial information will be studied in the following chapter—Chapter 8—in the context of determining the socially qualified individuals for incomplete profiles.

Chapter 8

Group Identification with Partial Information

In this chapter, we focus on a setting of group identification with partial information, based on our work [72]. Observe that given the full information of qualifications or disqualifications of these individuals, the socially qualified individuals with respect to all social rules mentioned in Section 6.1 can be calculated in polynomial time. However, in some real-world applications we are not able to obtain or access full information. For instance, if the number of individuals is extremely large (this happens often on online platforms), then it is not expected that every individual shows his opinion over every individual. Instead, every individual only qualifies or disqualifies a small number of individuals whom he knows well or whom he is particularly interested in. In addition, in some cases, even though the full qualifications and disqualifications exist, they cannot be entirely accessed by some specific people, say someone (or a company) who would like to predict the result. In this case, the one who wants to predict the result can only do so based on parts of the information. With the missing of some information, two significant questions arise: who have positive possibility to be socially qualified if the missing information is filled and who are definitely socially qualified regardless of the missing information? Moreover, how many complexity resources we need to achieve an answer to the above questions is of particular importance.

In this chapter, we study the complexity of two problems that capture the above two questions. More precisely, we study the *Possibly Qualified Individuals* problem (PQI) and the *Necessarily Qualified Individuals* problem (NQI). In both problems we are given a set of individuals N each of whom qualifies or disqualifies a subset of N, together with a subset $S \subseteq N$. The former problem asks whether there is an extension of these qualifications and disqualifications with respect to which all individuals in S are socially qualified, and the latter one asks whether all individuals in S are socially qualified with respect to every extension of these qualifications and disqualifications. Here, an extension means that every individual fills out potential gaps regarding individuals' qualifications, i.e., the individual determines a given qualification for individuals with previously undefined qualifications.

Investigating scenarios with incomplete information are relevant, especially from a practical perspective. The PQI/NQI problems considered here are a natural first step towards more complicated scenarios with incomplete information.

8.1 Problem Settings

In this section, we formally describe partial profiles. Intuitively, a partial profile is a profile with several 1s and 0s to be replaced with the symbol $*$, where $\varphi(a,b) = *$ means that whether a qualifies b or not is unknown, or a does not hold an opinion on the qualification of b. The formal definition is as follows.

A *partial profile* φ is a mapping $\varphi : N \times N \mapsto \{0,1,*\}$. For each $a \in N$ and $x \in \{0,1,*\}$, $x(a,\varphi)$ is the set of individuals $b \in N$ such that $\varphi(a,b) = x$, i.e., $x(a,\varphi) = \{b \in N : \ \varphi(a,b) = x\}$. A profile ϕ is an *extension* of a partial profile φ if and only if

1. for every $a,b \in N$ such that $\varphi(a,b) \in \{0,1\}$, it holds that $\phi(a,b) = \varphi(a,b)$; and
2. for every $a,b \in N$ such that $\varphi(a,b) = *$, it holds that $\phi(a,b) \in \{0,1\}$.

For a natural number $k \leq n := |N|$, a *k-profile* φ over N is a (complete) profile such that for every $a \in N$, it holds that $|1(a,\varphi)| = k$, i.e., each individual qualifies exactly k individuals in N. A partial profile φ is called an *k-partial profile* if $|1(a,\varphi)| \leq k$ for every $a \in N$. A k-profile ϕ is a *k-extension* of a k-partial profile φ if ϕ is an extension of φ.

Verify that a k-partial profile φ has a k-extension if and only if $|*(a,\varphi)| \geq k - |1(a,\varphi)|$ for every $a \in N$. Throughout this chapter, we consider only k-partial profiles that have k-extensions.

Before we continue, regard the following examples:

$$\varphi_1 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad \varphi_2 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & * & * & 0 \\ * & * & * & * \\ * & * & * & 1 \end{pmatrix}.$$

Observe that the complete profile φ_1 is a 2-profile as well as a 2-partial profile. Verify that $\phi_1 = \varphi_1$ is the only extension and the only 2-extension of φ_1.

φ_2 is a 3-partial profile and a (general) partial profile. Due to the first row with exactly three one entries, φ_2 is a 4-partial profile but not a 2-partial profile and has neither 2- nor 4-extensions. Possible extensions of φ_2 are given as follows:

$$\phi_2^1 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}, \quad \phi_2^2 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}.$$

Notice that ϕ_2^1 is an extension, but not a k-extension for any $k \leq 4$, whereas ϕ_2^2 is an extension and a 3-extension since each individual qualifies exactly three individuals. The first individual's opinions are uniquely determined as $*(a_1,\varphi_2) = \emptyset$. Likewise, although $*(a_2,\varphi_2) \neq \emptyset$, we know that a_2 definitely qualifies a_1, a_2, and a_3—provided that our extension is a 3-extension. About individual a_3, we only know that a_3 qualifies exactly three individuals, but not which ones. Individual a_4 certainly qualifies himself and possibly but not definitely qualifies two of the other three individuals (when focusing on extensions that are 3-profiles).

In this chapter, we study the complexity of the following problems.

f-POSSIBLY/f-NECESSARILY QUALIFIED INDIVIDUALS (f-PQI/f-NQI)	
Given:	A 3-tuple (N, φ, S) of a set N of individuals, a partial profile φ over N, and a nonempty subset $S \subseteq N$.
f-PQI:	Is there an extension ϕ of φ such that $S \subseteq f(\phi, N)$?
f-NQI:	Does $S \subseteq f(\phi, N)$ hold for every extension ϕ of φ?

In addition to the above problems, we also study f-k-PQI (f-k-NQI) where the input and question are similar to that of f-PQI (f-NQI) with only the following differences. First, in the input we require φ to be a k-partial profile rather than a partial profile. Second, in the question we replace "extension" with "k-extension". Notice that f-k-PQI (f-k-NQI) is not a special case of f-PQI (f-NQI) as one of the restrictions is on the solution space. Some polynomial-time results in this chapter still hold when s, t, or k are non-constant.

Throughout this chapter, we use $I = (N, \varphi, S \subseteq N)$ $(S \neq \emptyset)$ to denote the given instance in the problems we study for a social rule f, where $f \in \{f^{(s,t)}, f^{LSR}, f^{CSR}\}$, and do not state this explicitly in the proofs of the theorems. Furthermore, for better readability we make use of the diction "PQI/NQI for social rule f" instead of "f-PQI/f-NQI".

In some proofs, we borrow the notation in Chapter 7 and again let $y(a)$ denote the number of individuals in N qualifying a. Likewise, $n(a) := n - y(a)$ is the sum of all individuals disqualifying a ($y(a)$ and $n(a)$ are defined in the style of "yes" and "no").

In various proofs, we further let $S_i := \{a \in S : \varphi(a,a) = i\}$ $(i \in \{1, 0, *\})$.

8.2 Results

8.2.1 Unbounded Qualifications

In this section, we study PQI and NQI where each individual can qualify as many as up to n individuals. Our main results are polynomial-time algorithms for PQI and NQI for consent rules, liberal-start-respecting rule, and consensus-start-respecting rule.

Consider first consent rules. Our polynomial-time algorithms are based on the observation that if an individual a is socially qualified and someone else disqualifying a changes his opinion to qualifying a, then a is still qualified. In particular, when studying PQI, this observation enables us to safely reset the values of many $\varphi(a,b)$ with $\varphi(a,b) = *$ in advance to 1.

Theorem 8.1. *PQI and NQI for consent rules $f^{(s,t)}$ can be solved in $O(n^2)$ time, for all integers s and t.*

Proof. First regard PQI. For each individual $a \in S$, we may extend all entries $\varphi(b,a) = *$ $(b \in N \setminus \{a\})$ by $\phi(b,a) = 1$. Regardless of whether a qualifies or disqualifies himself, this is best possible for a: If a qualifies himself, this maximizes the number of qualifications for a. By contrast, in case a disqualifies himself, the number of disqualifications for a is thus minimized.

Note that changing the subprofile restricted to all rows and the columns according to individuals in S can be done in $O(n^2)$ and—more precisely—in $O(n|S|)$ time.

For individuals in $S_1 \cup S_0$, we obtain a complete subprofile by this (that is, we know all values $\phi(b,a)$ for each $b \in N$, $a \in S_1 \cup S_0$). For $a \in S_1$, we therefore check if $y(a) \geq s$ in the changed profile. If $a \in S_0$, we check whether or not $n(a) < t$ holds.

For $a \in S_*$, we simply try out both possibilities. First we let $\phi(a,a) = 1$ and check if $y(a) \geq s$ holds, provided that all individuals $b \neq a$ with $\varphi(b,a) = *$ qualify a. Afterwards, we set $\phi(a,a) = 0$ and check whether $n(a) < t$ holds. a is a possibly socially qualified individual if and only if at least one of these two possibilities makes a socially qualified.

In total, we can check in $O(|S|n)$ time if all individuals in S are socially qualified for at least one extension.

Now it remains to argue for the NQI problem. Independent from whether $a \in S_0$ or $a \in S_1$, the worst possible scenario for a is that all individuals b, with $b \neq a$ and $\varphi(b,a) = *$, disqualify a. Following this, we change the profile restricted to the columns according to individuals in S. For $a \in S_0 \cup S_1$, we simply need to verify whether or not $y(a) \geq s$ ($a \in S_1$) or $n(a) < t$ ($a \in S_0$) holds in the worst-case extension. For $a \in S_*$, a is necessarily socially qualified if and only if both setting $\phi(a,a) = 1$ and $\phi(a,a) = 0$ makes a socially qualified according to the $f^{(s,t)}$ rule. With roughly the same reasoning as for PQI (with the difference of replacing stars by zeroes instead of ones), our problem is in $O(|S|n)$, too. ❑

Now we turn our attention to the procedural rules f^{CSR} and f^{LSR}. Based on a similar observation to the one for consent rules, we develop polynomial-time algorithms for PQI and NQI for these two rules.

Theorem 8.2. *PQI and NQI for f^{LSR} and f^{CSR} can be solved in $O(n^2)$ time.*

Proof. Let $f \in \{f^{LSR}, f^{CSR}\}$ be any procedural rule. We first show that an individual a is a socially qualified individual under social rule f for at least one extension ϕ of φ if and only if a is socially qualified for the extension ϕ_1 where it holds $\phi_1(a,b) = 1$ for all entries with $\varphi(a,b) = *$. Note that the back direction trivially holds. Hence, we only have to show the first direction. By proving the first direction, we implicitly demonstrate that in case not all individuals in S are socially qualified under ϕ_1, there are no other extensions ϕ for which all individuals in S are socially qualified. The reason is as follows. Suppose that a is socially qualified according to $f \in \{f^{LSR}, f^{CSR}\}$ under ϕ. Then there are $r+1$ pairwise different individuals $a^{(0)}, a^{(1)}, \ldots, a^{(r)}$ in N ($r \in \mathbb{N}_0$) such that $a^{(0)}$ is in the starting set, $a^{(r)} = a$, and $\phi(a^{(j)}, a^{(j+1)}) = 1$ holds for every $j \in \{0, 1, \ldots, r-1\}$ (possibly, it holds $r = 0$ which means that a is in the starting set). Note that this sequence still consists when we replace all entries with $\varphi(a,b) = *$ and $\phi(a,b) = 0$ by $\phi_1(a,b) = 1$ (according to this, we obtain the profile ϕ_1). We have $\phi_1(a^{(j)}, a^{(j+1)}) = 1$ for each $j \in \{0, 1, \ldots, r-1\}$ and a is socially qualified in the profile ϕ_1 as well.

Next we define ϕ_0 as an extension of φ, where we set $\phi_0(a,b) = 0$ each time $\varphi(a,b) = *$ holds. We claim that a is socially qualified under the profile ϕ_0 if and only if a is socially qualified for every extension ϕ of φ. Again, the opposite direction is trivial. It suffices to show the first direction. As a is socially qualified under ϕ_0, there exist again $r+1$ pairwise different individuals $a^{(0)}, a^{(1)}, \ldots, a^{(r)}$ in N ($r \in \mathbb{N}_0$) such that $a^{(0)}$ is in the starting set, $a^{(r)} = a$, and $\phi_0(a^{(j)}, a^{(j+1)}) = 1$ holds for every $j \in \{0, 1, \ldots, r-1\}$. Observe that each extension $\phi \neq \phi_0$ can be won from ϕ_0 by turning some zeroes into ones, where $\varphi(a,b) = *$ holds in the partial profile. In contrast, ϕ_0 contains the largest possible number of zeroes. Consequently, it holds $\phi(a^{(j)}, a^{(j+1)}) = 1$ for each j for any other extension ϕ, and a is still socially qualified.

ϕ_1 can be regarded as the best-case profile as all undetermined entries are extended to qualifications, whereas in ϕ_0, all unsure qualifications turn out to be disqualifications.

According to the reasoning above, we only have to check whether all individuals in S are socially qualified under the extensions ϕ_1 (for PQI) and ϕ_0 (for NQI), respectively.

Since computing the optimistic and pessimistic profiles can be done in $O(n^2)$ time (scanning all entries in the matrix and changing them, if necessary) and calculating the set of socially qualified individuals in both social rules is possible in $O(n^2)$ as well, our overall problem is in $O(n^2)$. ❑

8.2.2 k-Partial Profiles

In this section, we study two variants of PQI and NQI, namely k-PQI and k-NQI. In particular, in these two variants every individual must qualify exactly k individuals in extensions of the given partial profile. We have seen in the previous section that PQI and NQI are polynomial-time solvable for all social rules considered. The intuition is that in all cases, maximizing the number of individuals who qualify an individual a increases the possibility of a to be socially qualified in a given extension, and maximizing the number of individuals who disqualify a decreases the possibility of a to be socially qualified. As such, in PQI/NQI we generally replace $*$ with $1/0$. However, if every individual is allowed to qualify exactly k individuals, we have to carefully replace $*$ with 1 or 0.

We prove that k-PQI and k-NQI for consent rules are polynomial-time solvable too. However, the polynomial-time algorithms studied in this section are not trivial generalizations of the ones studied in the previous section. In fact, we derive different algorithms for consent rules with different consent quotes s and t. Moreover, we obtain some of the polynomial-time algorithms only when k and t are both constants. However, the polynomial-time algorithms studied in the previous section hold for all integers s and t. For the two procedural social rules f^{CSR} and f^{LSR}, we prove that 1-PQI and k-NQI ($k \in \mathbb{N}$) are polynomial-time solvable. However, if k increases just by one, we show that k-PQI for both procedural rules becomes NP-hard. Hence, we obtain complexity dichotomy results for k-PQI and k-NQI for f^{CSR} and f^{LSR} with respect to the values of k.

An observation that is useful in deriving the polynomial-time algorithms is as follows: if there is an individual $a \in N$ such that $|1(a,\varphi)| = k$, then in each k-extension ϕ of φ it must be that $\phi(a,b) = 0$ for every $b \in *(a,\varphi)$. In addition, if $|*(a,\varphi)| = k - |1(a,\varphi)|$ for an individual $a \in N$, in every k-extension ϕ of φ it must be that $\phi(a,b) = 1$ for every $b \in *(a,\varphi)$.

We consider first k-PQI and k-NQI for consent rules. Our first result is a polynomial-time algorithm for k-NQI for consent rules $f^{(s,t)}$. Moreover, the polynomial-time solvability holds regardless of the values of s and t.

Theorem 8.3. *k-NQI for all consent rules $f^{(s,t)}$ can be solved in $O(n^2)$ time.*

Proof. We develop a polynomial-time algorithm as follows. First, the algorithm calculates $|1(a,\varphi)|$ and $|*(a,\varphi)|$ for all $a \in N$. This can be done in $O(n^2)$ time. Then, the algorithm breaks down I into $|S|$ subinstances, each of which takes as input I and an individual $a \in S$, and asks if there is a k-extension ϕ of φ such that $a \notin f^{(s,t)}(\phi, N)$. Observe that I is a NO-instance if and only if at least one of the subinstances is a YES-instance. Let $I' = (I, a \in S)$ be a subinstance. We show how to solve I' in polynomial time, by distinguishing between the following cases. Observe that, no matter if a qualifies or disqualifies himself, a k-extension with more individuals disqualifying a is more likely to prevent a from being a socially qualified individual.

- $\varphi(a,a) \in \{0,1\}$. Due to the above observation, we only need to check if a is socially qualified under an extension with as many individuals disqualifying a as possible. To this end, we have to call the following procedure:

  For every $b \in N$ such that $a \in *(b,\varphi)$, if $|*(b,\varphi)| > k - |1(b,\varphi)|$, reset $\varphi(b,a) = 0$; [1] otherwise, reset $\varphi(b,a) = 1$. This can be done in $O(n)$ time. If $\varphi(a,a) = 1$ and $|\{b \in N : \varphi(b,a) = 1\}| < s$, or $\varphi(a,a) = 0$ and $|\{b \in N : \varphi(b,a) = 0\}| \geq t$ after doing so, I' is a YES-instance. Otherwise, I' is a NO-instance. We can check this in $O(n)$ time.

- $\varphi(a,a) = *$. Assume that $|*(a,\varphi)| > k - |1(a,\varphi)|$ (and also $|1(a,\varphi)| < k$) since otherwise a qualifies (disqualifies) himself in all k-extensions of φ. In this case, we can reset $\varphi(a,a) = 1$ ($\varphi(a,a) = 0$) and solve I' by calling the procedure in the first case. We deal with the second case with this assumption as follows. First, for every $b \in N \setminus \{a\}$ such that $a \in *(b,\varphi)$, if $|*(b,\varphi)| > k - |1(b,\varphi)|$, reset $\varphi(b,a) = 0$; otherwise, reset $\varphi(b,a) = 1$. After doing so, we compare $s-1$ with the number of individuals qualifying a. In particular, if $|\{b \in N : \varphi(b,a) = 1\}| \leq s-1$ after resetting $\varphi(a,a) = 1$, a is not socially qualified in every k-extension of φ. Hence, we can conclude that I' is a YES-instance. If, however, $|\{b \in N : \varphi(b,a) = 1\}| \geq s$, we reset $\varphi(a,a) = 0$. Then, if $|\{b \in N : \varphi(b,a) = 0\}| \geq t$, a is not socially qualified in every k-extension of φ, implying that I' is a YES-instance as well. In all other cases, I' is a NO-instance. The above procedure can be done in $O(n)$ time.

As we have at most $|S| \leq n$ subinstances, the whole running time of the algorithm is bounded by $O(n^2) + n \cdot O(n) = O(n^2)$, where the first $O(n^2)$ is the time for calculating $|1(a,\varphi)|$ and $|*(a,\varphi)|$ for all $a \in N$. ❑

Now we consider k-PQI for consent rules. We start with a special case of k-PQI where in the input profile φ, it holds that $\varphi(a,a) \in \{0,1\}$ for every $a \in S$. We denote this special case by k-PQI-S and show that this problem for all consent rules $f^{(s,t)}$ is polynomial-time solvable even for k,s,t being non-constants. This algorithm will be used to develop polynomial-time algorithms for k-PQI for some consent rules later.

For an individual $a \in N$, let $x^{-1}(a,\varphi) = \{b \in N : \varphi(b,a) = x\}$, where $x \in \{0,1,*\}$. Observe that $1^{-1}(a,\varphi) = y(a)$ and $0^{-1}(a,\varphi) = n(a)$. We introduce this additional formalism to emphasize the dependence on a particular k-partial profile.

Lemma 8.4. *k-PQI-S for consent rules can be solved in $O(n^3)$ time.*

Proof. Suppose that φ is a k-partial profile with $\varphi(a,a) \in \{0,1\}$ for every $a \in S$. We solve the problem in polynomial-time by reducing it to the MAXIMUM FLOW problem and create the following network.

For every $a \in N$, we create one vertex $v(a)$. Moreover, for every $b \in S$ we further define one vertex $u(b)$. Finally, we create a source vertex x and a sink vertex y. The edges and capacities of the edges are as follows. First, there is an edge from the source x to every $v(a)$ with capacity $cap((x,v(a))) = k - |1(a,\varphi)|$, indicating that a can further qualify at most $k - |1(a,\varphi)|$ individuals in S, apart from individuals definitely qualified by a. Second, there is an edge from a vertex $v(a), a \in N$,

[1] Actually, we could also write ϕ as the resetting of φ means that we extend φ. For the sake of readability and because we only extend parts of the profile, we often argue with φ instead of ϕ throughout our proofs.

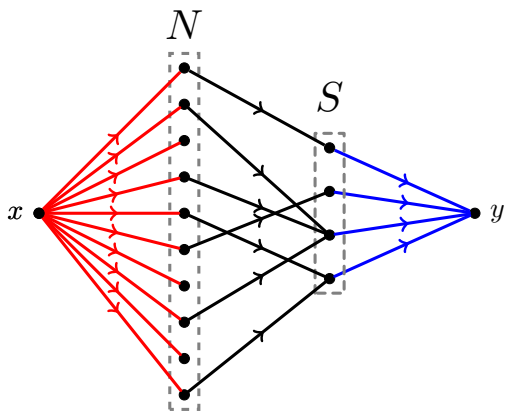

Figure 8.1: An illustration of the construction of the network in the proof of Lemma 8.4. Each red edge from x to a vertex $v(a)$ corresponding to an individual $a \in N$ has capacity $k - |1(a,\varphi)|$. Each blue edge from a vertex $u(b)$ corresponding to an individual $b \in S$ to y has capacity $\max(0, s' - |1^{-1}(b,\varphi)|)$. A black edge from $v(a), a \in N$ to $u(b), b \in S$ means that $\varphi(a,b) = *$. Moreover, each black edge in the middle has capacity 1.

to a vertex $u(b), b \in S$, with capacity 1 if and only if $\varphi(a,b) = *$, indicating that it is possible (but, in general, not definite [2]) to let a qualify b in a k-extension of φ. Finally, there is an edge from every vertex $u(b), b \in S$ to the sink y with capacity $cap((u(b),y)) = \max(0, s' - |1^{-1}(b,\varphi)|)$, where $s' = s$ if $\varphi(b,b) = 1$ and $s' = n - t + 1$ if $\varphi(b,b) = 0$. The capacity of the edge from $u(b)$ to y indicates how many qualifications are still needed to make b socially qualified. See Figure 8.1 for an illustration of the network.

We argue that our network yields an integral flow of size $\sum_{b \in S} cap((u(b),y))$ if and only if I is a YES-instance.

($\Leftarrow$): Assume that there is a k-extension ϕ of φ under which all individuals in S are socially qualified with respect to $f^{(s,t)}$. Consider the following flow. First, the flow on each edge from $u(b), b \in S$ to y is $cap((u(b),y))$. Regard now the flows on the edges in the middle. As each individual $b \in S$ is socially qualified in ϕ, the number of individuals a such that $\varphi(a,b) = *$ and $\phi(a,b) = 1$ is at least $\max(0, s' - |1^{-1}(b,\varphi)|) = cap((u(b),y))$, where s' is defined as above. Then, from these individuals, we select any arbitrary $cap((u(b),y))$ individuals $a \in N$. Moreover, for each such selected individual a the flow on the edge from $v(a)$ to $u(b)$ is 1. Finally, the flow on each edge from x to every $v(a), a \in N$ is the sum of flows leaving $v(a)$. Note that there can be at most $k - |1(a,\varphi)|$ individuals $b \in S$ such that $\varphi(a,b) = *$ and $\phi(a,b) = 1$. Hence, the flow on each edge from x to $v(a), a \in N$ does not exceed the capacity of the edge. The flows on all remaining edges are 0. Verify that the size of the flow is $\sum_{b \in S} cap((u(b),y))$.

($\Rightarrow$): Assume that our network has an integral flow of size $\sum_{a \in S} cap((u(a),y))$. We can find a k-extension ϕ of φ under which all individuals in S are socially qualified as follows. Consider the k-partial profile ϕ obtained from φ by resetting $\varphi(a,b) = 1$ for every edge from a vertex $v(a), a \in N$ to a vertex $u(b), b \in S$ with flow 1. Hence, for each $b \in S$ with $\varphi(b,b) = 1$, there are at least

[2] Whenever $\varphi(a,b) = *$, a definitely qualifies b only if $|1(a,\varphi)| + |*(a,\varphi)| = k$.

$|1^{-1}(b,\varphi)|+cap((u(b),y)) \geq s$ individuals qualifying b with respect to ϕ. In addition, for each $b \in S$ with $\varphi(b,b) = 0$, there are at least $|1^{-1}(b,\varphi)|+cap((u(b),y)) \geq n-t+1$ individuals qualifying b. This means that there are at most $t-1$ individuals disqualifying b with respect to ϕ. Then, due to the definition of $f^{(s,t)}$, any k-extension ϕ of φ is a solution of I.

It remains to analyze the running time of the algorithm. It was recently shown by Orlin [136] that the MAXIMUM FLOW problem is solvable in $O(p \cdot q)$ time, where p is the number of edges and q is the number of vertices in the network. As the network constructed above has $O(n^2)$ edges and $O(n)$ vertices, the running time of our algorithm is bounded by $O(n^3)$. ❑

Armed with Lemma 8.4, we are ready to show our polynomial-time algorithms for k-PQI for consent rules. We start with some results for k-PQI for consent rules with $(s,t) = (1,2)$ or $t = 1$, and unbounded values of k.

Theorem 8.5. *k-PQI for $f^{(1,1)}$ and for $f^{(1,2)}$ can be solved in $O(n^2)$ time. For $f^{(s,1)}$, k-PQI is in $O(n^3)$.*

Proof. We develop polynomial-time algorithms with the corresponding running times as stated in the theorem as follows.

$f^{(1,1)}$: If there is an individual $a \in S$ such that $\varphi(a,a) = 0$, I is a NO-instance. This can be checked in $O(|S|) = O(n)$ time. In addition, if there is an $a \in S$ such that $\varphi(a,a) = *$ and $|1(a,\varphi)| = k$, the instance is a NO-instance as well since in all k-extensions of φ, individual a disqualifies himself. If such an individual a does not exist, the instance must be a YES-instance. As it takes $O(n)$ time to calculate $|1(a,\varphi)|$ for each $a \in S$, the algorithm terminates in $O(n)+|S| \cdot O(n) = O(n^2)$ time.

$f^{(1,2)}$: If there exist $a \in S$ and $b \in N \setminus \{a\}$ such that $\varphi(a,a) = \varphi(b,a) = 0$, we conclude that I is a NO-instance. In addition, if there exist $a \in S$ and $b \in N \setminus \{a\}$ such that $\varphi(a,a) = *$ and $\varphi(b,a) = 0$, we reset $\varphi(a,a) = 1$. If after the reset $|1(a,\varphi)| > k$, we conclude that I is a NO-instance. We deal with the remaining cases as follows. It is clear that if $\varphi(a,a) = 1$ for some $a \in S$, or $\varphi(a,a) = 0$ and $\varphi(b,a) = 1$ for all $b \in N \setminus \{a\}$, then a is socially qualified in every k-extension of φ. We need only to focus on other individuals in S. Let S_0^* be the set of individuals $a \in S$ such that $\varphi(a,a) = 0$, there exists some $b \in N \setminus \{a\}$ such that $\varphi(b,a) = *$, and it holds $\varphi(b,a) \neq 0$ for each $b \in N \setminus \{a\}$. We maintain the set S_0^* throughout the algorithm. For every $a \in S_0^*$, we have to reset $\varphi(b,a) = 1$ for all $b \in N$ such that $\varphi(b,a) = *$ since otherwise a would not be socially qualified. Due to this, we derive a procedure to deal with the remaining cases as follows. While $S_0^* \neq \emptyset$, for every $a \in S_0^*$ and $b \in N$ such that $\varphi(b,a) = *$, reset $\varphi(b,a) = 1$. If $k < |1(b,\varphi)|$ after resetting $\varphi(b,a) = 1$, the procedure immediately returns "NO".

After the while loop, for all $b \in N$ with $k = |1(b,\varphi)|$, we do the following:

(1) reset $\varphi(b,a) = 0$ for all $a \in *(b,\varphi)$ (note that after this adjustment of the profile, there might be some $a \in S$ with $\varphi(a,a) = \varphi(b,a) = 0$ for some $b \in N \setminus \{a\}$; in this case, we reject.), (2) update S_0^*; [3] and (3) go back to the while loop if $S_0^* \neq \emptyset$.

If no conclusion is drawn after the while loops terminate and after step (1) directly following the last iteration, we can conclude that I is a YES-instance. The reason is as follows. For every $a \in S$ with $\varphi(a,a) = *$ (note that there is no individual left in S_0^* and there has not been any socially

[3] After step (1), it is possible that an individual $a \in S$ with $\varphi(a,a) = *$ at the beginning currently disqualifies himself. If it holds $\varphi(b,a) \in \{1,*\}$ for each $b \in N \setminus \{a\}$, a is presently belonging to S_0^*.

disqualified individual in S up to the present), we know that $|1(a,\varphi)| < k$ holds; therefore, we can reset $\varphi(a,a) = 1$ to make a socially qualified. Notice that there are no individuals $a \in S$ left with $\varphi(a,a) = *$ and $|1(a,\varphi)| = k$. For these individuals, we would have reset $\varphi(a,a) = 0$ in step (1) directly following the while loop and either there would have been a contradiction or we would have run again the while loop for a after the updating in step (3).

Observe that the above algorithm runs in $O(n^2)$ time.

$f^{(s,1)}$, $s > 1$: If there is an individual $a \in S$ such that $\varphi(a,a) = 0$, or $\varphi(a,a) = *$ and $k = |1(a,\varphi)|$, then I is a NO-instance since a is not socially qualified in any k-extension of φ. As $|1(a,\varphi)|$ can be calculated in $O(n)$ time for a given a, this can be checked in $O(n^2)$ time. Assume that no such individuals a as discussed above exist. Then, if there is an $a \in S$ such that $\varphi(a,a) = *$, we can safely reset $\varphi(a,a) = 1$: if there is a k-extension of φ under which all individuals in S are socially qualified, then a must qualify himself in the k-extension. This can be done in $O(|S|) = O(n)$ time. Hence, we have now that $\varphi(a,a) = 1$ for every $a \in S$. Then, due to Lemma 8.4, we can solve the instance in $O(n^3)$ time. Notice that the whole running time of the algorithm is bounded by $O(n) + O(n^3) = O(n^3)$. ❑

We have shown that if $(s,t) = (1,2)$ or $t = 1$, k-PQI for $f^{(s,t)}$ is polynomial-time solvable even when k (and s, when regarding $f^{(s,1)}$) is not a constant. In the following, we continue to show some polynomial-time algorithms based on Lemma 8.4. However, these algorithms are based on the assumption that k and t are constants. Before showing the result, let us first study a property, as summarized in the following lemma. In general, it states that in any k-profile, the number of socially qualified individuals disqualifying themselves is bounded by $k+t-1$. Hence, due to this property, we can enumerate all individuals $a \in S$ with $\varphi(a,a) = *$ who disqualify themselves in a solution in $O(n^{k+t-1})$ time. If both k and t are constants, this can be done in polynomial time. Moreover, we can solve k-PQI in polynomial-time based on Lemma 8.4.

Lemma 8.6. *Let φ be a k-profile over N. Let A be the set of socially qualified individuals with respect to the consent rule $f^{(s,t)}$ who disqualify themselves, i.e., $A = \{a \in f^{(s,t)}(\varphi,N) :\ \varphi(a,a) = 0\}$. Then, $|A| \leq k+t-1$.*

Proof. Consider the profile φ restricted to A. Let x be the number of qualifications in the subprofile, i.e., $x = |\{\varphi(a,b) :\ a,b \in A, \varphi(a,b) = 1\}|$. As each individual qualifies k individuals, we have that $x \leq k \cdot |A|$. On the other hand, as each individual in A is socially qualified, for each $a \in A$ there are at most $t-1$ individuals in A disqualifying a and consequently at least $|A|-t+1$ individuals in A qualifying a. It follows that $x \geq |A| \cdot (|A|-t+1)$. In summary, we have that $|A| \cdot (|A|-t+1) \leq k \cdot |A|$. It directly follows that $|A| \leq k+t-1$. ❑

Now we are ready to show an algorithm to solve k-PQI for all consent rules.

Theorem 8.7. *k-PQI for $f^{(s,t)}$ can be solved in $O(n^{(t+k+2)})$ time.*

Proof. According to Lemma 8.6, if I is a YES-instance, then for any solution ϕ of I, it must be that $|\{a \in S :\ \phi(a,a) = 0\}| \leq k+t-1$. Due to this, we can enumerate all subsets A of S_* such that all individuals in A disqualify themselves and all individuals in $S_* \setminus A$ qualify themselves in a solution ϕ of I, i.e., $A = \{a \in S_* :\ \phi(a,a) = 0\}$. Then, for each enumerated A, we extend φ by resetting $\varphi(a,a) = 0$ for all individuals $a \in A$ and resetting $\varphi(a,a) = 1$ for all individuals

$a \in S_* \setminus A$ (if φ is not a k-partial profile after the resetting, we discard this enumeration). So, each enumeration corresponds to an instance of k-PQI-S. Observe that I is a YES-instance if and only if at least one of the enumerations corresponds to a YES-instance. Due to Lemma 8.4, k-PQI-S can be solved in $O(n^3)$ time. As we have in total at most $\binom{|S|}{0} + \binom{|S|}{1} + \ldots + \binom{|S|}{k+t-1} \leq \binom{n}{0} + \binom{n}{1} + \ldots + \binom{n}{k+t-1} = O(n^0) + O(n^1) + \ldots + O(n^{(k+t-1)}) = O(n^{(k+t-1)})$ enumerations, we can solve I in $O(n^3) \cdot O(n^{(k+t-1)}) = O(n^{(k+t+2)})$ time. [4] ❑

According to the above theorem, if both k and t are constants, we can solve k-PQI for $f^{(s,t)}$ in polynomial-time. One may wonder if we could get a similar result but with the assumption that s is a constant. As a matter of fact, if we assume that $n-k$ is a constant, i.e., each individual is allowed to disqualify a constant number of individuals, we could obtain a similar lemma to Lemma 8.6. Based on the lemma, we could obtain an algorithm with running time $O(n^{(n-k+s+2)})$. In addition, by utilizing Lemma 8.4 we could derive an algorithm with running time $O(2^{|S|} \cdot n^3)$: enumerate all possible values of $\varphi(a,a)$ for all $a \in S$ with $\varphi(a,a) = *$. Observe that there are at most $2^{|S|}$ enumerations and we can solve each enumerated case in $O(n^3)$ time according to Lemma 8.4.

In the remainder, we study k-PQI and k-NQI for the two procedural rules f^{CSR} and f^{LSR}. We prove that the values of k have a significant impact on the complexity of these two problems. First, we show that 1-PQI and 1-NQI for f^{LSR} and f^{CSR} are polynomial-time solvable, based on the following lemma.

Lemma 8.8. *Let $X \in \{CSR, LSR\}$ and ϕ a 1-extension of φ. Then, an individual $a \in N$ is socially qualified in ϕ with respect to f^X if and only if a is in the initial set of socially qualified individuals in ϕ, i.e., $a \in K_0^X(\phi, N)$.*

Proof. As ϕ is a 1-extension of φ, every individual in N qualifies exactly one individual. Then, due to the definition of f^X, an individual in the initial set of ϕ with respect to f^X does not qualify anyone not in the initial set. It directly follows that $f^X(\phi, N) = K_0^X(\phi, N)$, i.e., the socially qualified individuals with respect to f^X and ϕ are exactly the individuals in the initial set of socially qualified individuals. The lemma follows. ❑

Based on the above lemma, we are able to show our results.

Theorem 8.9. *1-PQI and 1-NQI for f^{CSR} and f^{LSR} can be solved in $O(n^2)$ time.*

Proof. Due to Lemma 8.8, 1-PQI for f^{CSR} (f^{LSR}) consists in determining whether there is a 1-extension of φ where every $a \in S$ is in the initial set K_0^{CSR} (K_0^{LSR}) of socially qualified individuals. We develop polynomial-time algorithms for 1-PQI as follows.

f^{LSR}: Return "NO" if and only if there is an $a \in S$ such that $\varphi(a,a) = 0$, or $\varphi(a,a) = *$ and $k = 1 = |1(a,\varphi)|$. This can be done in $O(n^2)$ time. Notice that if $\varphi(a,a) = *$ for some $a \in S$ and $k = 1 > |1(a,\varphi)| = 0$, we can safely reset $\varphi(a,a) = 1$ without changing the answer to the instance.

[4] It may occur that α individuals in S definitely disqualify themselves. Hence, it suffices to consider extensions ϕ for which at most $k+t-1-\alpha$ individuals in S_* disqualify themselves.

f^{CSR}: Note that in every 1-extension, there can be at most one individual in the initial set of socially qualified individual with respect to f^{CSR}. Hence, due to Lemma 8.8, we can immediately conclude that I is a NO-instance if $|S| \geq 2$. Assume now that $S = \{a\}$. Again, due to Lemma 8.8, if there is an individual $b \in N$ such that $\varphi(b,a) = 0$, or $\varphi(b,a) = *$ and $k = 1 = |1(b,\varphi)|$, I is a NO-instance; otherwise, I is a YES-instance as we can find a solution obtained from φ by resetting $\varphi(b,a) = 1$ for every $\varphi(b,a) = *$ in advance. In the worst case, we need to calculate $|1(b,\varphi)|$ for all $b \in N$. As $|1(b,\varphi)|$ can be calculated in $O(n)$ time for each $b \in N$, the algorithm terminates in $n \cdot O(n) = O(n^2)$ time.

Now we turn our attention to 1-NQI for f^{CSR} (f^{LSR}). To solve the problem, it suffices to determine if there is an $a \in S$ and a 1-extension ϕ of φ such that $a \notin f^{CSR}(\phi,N)$ ($f^{LSR}(\phi,N)$). Due to Lemma 8.8, this is equivalent to determining if there is a 1-extension of φ such that a is not in the initial set of socially qualified individuals. We accept if this is impossible for all individuals in S. Hence, we can use similar algorithms to the ones for 1-PQI to solve the instance here.

For f^{LSR}, we accept if and only if for all $a \in S$, it holds either $\varphi(a,a) = 1$ or $\varphi(a,a) = *$ and $|0(a,\varphi)| = n-1$.

For f^{CSR}, we accept if and only if $S = \{a\}$ (that is, S is a singleton) and for each $b \in N$ it holds either $\varphi(b,a) = 1$ or $\varphi(b,a) = *$ and $|0(b,\varphi)| = n-1$.

Note that both conditions can be checked in $O(n^2)$ time. ❑

In contrast to the polynomial-time solvability of 1-PQI and 1-NQI, we show that if the individuals are allowed to qualify one more individual, k-PQI for f^{CSR} and f^{LSR} become NP-hard, i.e., k-PQI for f^{LSR} and f^{CSR} are NP-hard for every constant $k \geq 2$. Our proofs are based on reductions from the HAMILTONIAN PATH problem which is known to be NP-hard [90]. We first show the NP-hardness of k-PQI for f^{LSR}.

Theorem 8.10. *k-PQI for f^{LSR} is* NP*-complete for every constant $k \geq 2$.*

Proof. Let $G = (\mathscr{V},\mathscr{E})$ be a given instance of the HAMILTONIAN PATH problem. We create an instance $I = (N,\varphi,S \subseteq N)$ of f^{LSR}-k-PQI as follows.

For each vertex $u \in \mathscr{V}$, we create an individual $a(u)$. In addition, we have k individuals $b_0,\ldots,b_{k-1}$. We define $S = \{a(u) : u \in \mathscr{V}\}$. Hence, $N = S \cup \{b_0,\ldots,b_{k-1}\}$. The k-partial profile φ is defined as follows.

- $\varphi(a,a) = 0$ for every $a \in S$.
- $\varphi(a,b_i) = 1$ for every $a \in N$ and $0 \leq i \leq k-2$.
- $\varphi(b_i,b_{k-1}) = 1$ for every $1 \leq i \leq k-1$ and $\varphi(a,b_{k-1}) = 0$ for every $a \in S \cup \{b_0\}$.
- $\varphi(b_0,a(u)) = *$ for every $a(u) \in S$.
- $\varphi(b_i,a(u)) = 0$ for every $1 \leq i \leq k-1$ and $a(u) \in S$.
- For $a(w),a(u) \in S$, if $(w,u) \in \mathscr{E}$, then $\varphi(a(w),a(u)) = *$; otherwise, $\varphi(a(w),a(u)) = 0$.

It remains to prove the correctness of the reduction. We claim that there exists a Hamiltonian path in G if and only if there is a k-extension of φ such that all individuals in S are socially qualified.

($\Rightarrow$): Assume that there is a Hamiltonian path $(u_1,\ldots,u_q)$ in G, where $q = |\mathscr{V}|$. Consider the k-extension ϕ of φ obtained from φ by resetting the values of $*$ entries as follows.

- Set $\phi(b_0, a(u_1)) = 1$.
- For every $1 \leq i \leq q-1$, set $\phi(a(u_i), a(u_{i+1})) = 1$.
- Set $\phi(a(u_q), a(u_i)) = 1$, where u_i is any arbitrary outneighbor of u_q in G.
- Reset $\phi(a(u_i), a(u_j)) = 0$ for all entries not defined above.

In this extension, b_0 is socially qualified. According to the definition of f^{LSR}, $a(u_1)$ is also socially qualified as b_0 qualifies $a(u_1)$ in ϕ. Moreover, if $a(u_i)$ is socially qualified, so is $a(u_{i+1})$ for every $i \in \{1, 2, \ldots, q-1\}$. This implies that all individuals in S are socially qualified in ϕ. Hence, I is a YES-instance.

($\Leftarrow$): Let ϕ be a k-extension of φ with respect to which all individuals in S are socially qualified. As each individual $a(u) \in S$ qualifies the $k-1$ individuals $b_0, b_1, \ldots, b_{k-2}$, $a(u)$ qualifies exactly one more individual $a(w) \in S$. Moreover, b_0 qualifies exactly one individual in S. Hence, from b_0, we can find a sequence of individuals $a(u_1), a(u_2), \ldots, a(u_q)$ such that b_0 qualifies $a(u_1)$ and $a(u_i)$ qualifies $a(u_{i+1})$ for every $1 \leq i \leq q-1$. Due to the construction, $(u_1, u_2, \ldots, u_q)$ is a Hamiltonian path in G. ❑

We can prove the NP-hardness of k-PQI for f^{CSR} by a similar reduction.

Theorem 8.11. *k-PQI for f^{CSR} is* NP*-complete for every constant $k \geq 2$.*

Proof. Let us first show NP-hardness of 2-PQI for f^{CSR}. We will show later how to adjust the reduction to the case $k \geq 3$. Let $G = (\mathscr{V}, \mathscr{E})$ be a given instance of the HAMILTONIAN PATH problem. We create an instance $I = (N, \varphi, S \subseteq N)$ of f^{CSR}-k-PQI as follows.

For each vertex $u \in \mathscr{V}$, we create an individual $a(u)$. In addition, we have an individual b. We define $S = \{a(u) : u \in \mathscr{V}\}$. The 2-partial profile φ is defined as follows.

- $\varphi(a(u), b) = 1$ and $\varphi(b, a(u)) = *$ for every $a(u) \in S$.
- $\varphi(b, b) = 1$.
- $\varphi(a(u), a(u)) = 0$ for every $a(u) \in S$.
- For distinct $a(w), a(u) \in S$, if $(w, u) \in \mathscr{E}$, then $\varphi(a(w), a(u)) = *$; otherwise, $\varphi(a(w), a(u)) = 0$.

This completes the construction. Notice that b is socially qualified in every 2-extension ϕ of φ. It remains to prove the correctness of the reduction. We claim that there is a Hamiltonian path in G if and only if our instance of f^{CSR}-k-PQI is a YES-instance.

($\Rightarrow$): Assume that there is a Hamiltonian path $(u_1, \ldots, u_q)$ in G, where $q = |\mathscr{V}|$. Consider the 2-extension ϕ of φ obtained from φ by resetting the values of $*$ entries as follows.

- $\phi(b, a(u_1)) = 1$.
- For every $1 \leq i \leq q-1$, set $\phi(a(u_i), a(u_{i+1})) = 1$.
- Set $\phi(a(u_q), a(u_i)) = 1$, for an arbitrary u_i with $(u_q, u_i) \in \mathscr{E}$.
- Set $\phi(a(u_i), a(u_j)) = 0$ for all $*$ entries not defined above.

In this extension, b is qualified by all individuals. According to the definition of f^{CSR}, b is socially qualified. Moreover, as b qualifies $a(u_1)$, $a(u_1)$ is also socially qualified. Furthermore, if $a(u_i)$ is socially qualified, so is $a(u_{i+1})$ for every $i \in \{1,...,q-1\}$. This means that all individuals in S are socially qualified in ϕ. Hence, I is a YES-instance.

($\Leftarrow$): Let ϕ be a 2-extension of φ with respect to which all individuals in S are socially qualified. As each individual $a(u) \in S$ definitely qualifies the individual b in φ, $a(u)$ qualifies exactly one more individual $a(w) \in S$. Moreover, b qualifies exactly one individual in S. Hence, from b, we can find a sequence of individuals $a(u_1), a(u_2), \ldots, a(u_q)$ such that b qualifies $a(u_1)$ and $a(u_i)$ qualifies $a(u_{i+1})$ for every $1 \leq i \leq q-1$. Due to the construction, $(u_1, u_2, \ldots, u_q)$ is a Hamiltonian path in G. This completes the proof.

The reduction for the case $k \geq 3$ can be obtained from the above one by a slight modification. In particular, we create further $k-1$ dummy individuals $b_1, \ldots, b_{k-1}$. Let B be the set of these individuals. So, we have $q+k$ individuals and $N = S \dot{\cup} \{b\} \dot{\cup} B$ now. The profile restricted to $S \cup \{b\}$ is defined exactly as in the above reduction. The values of the remaining entries are as follows.

- Every individual in $S \cup \{b\}$ qualifies the individuals $b_1, ..., b_{k-2}$.
- Every individual of $\{b_1, \ldots, b_{k-1}\}$ qualifies the individuals in $\{b\} \cup B$.
- $\varphi(a, a') = 0$ for all $a, a' \in N$ for which $\varphi(a, a')$ has not been defined yet.

The point is that no individual in B qualifies an individual in S. Moreover, each individual in $S \cup \{b\}$ is able to qualify at most one individual in S in any k-extension of φ, just as in the above reduction for 2-PQI. ❑

In the k-partial profiles constructed in the proofs of Theorem 8.10 and Theorem 8.11, all individuals in S disqualify themselves. Hence, we in fact proved that k-PQI-S for both f^{LSR} and f^{CSR} with $k \geq 2$ is NP-complete.

In contrast, k-NQI is easy for both procedural rules and arbitrary $k \in \mathbb{N}$.

Theorem 8.12. *f^{CSR}-k-NQI and f^{LSR}-k-NQI are in $O(n^4)$ for every $k \geq 2$.*

Proof. W.l.o.g., N contains more than k individuals (hence, $n > 2$ holds as $k \geq 2$) as otherwise all individuals are trivially socially qualified for each k-extension. Let us start with the consensus-start respecting rule. In a first step, our algorithm checks whether there is a k-extension ϕ of φ such that the starting set $K_0^C(\phi, N)$ is empty. In that case, the answer to our problem is NO as there are no socially qualified individuals at all for such k-extensions ϕ. After doing so, in case the starting set is non-empty for all k-extensions ϕ, our algorithm checks in a second step for each individual $b \in S$ whether there are k-extensions for which b is socially disqualified. We accept if and only if making b socially disqualified is impossible for each $b \in S$.

The starting set according to a given k-extension ϕ is empty if and only if for each individual $a \in N$ there is an individual $a' \in N$ with $\phi(a', a) = 0$. In other words, each individual a is disqualified by at least one individual in the (complete) k-profile ϕ extending φ. Note that if $\varphi(a', a) = 0$ holds for $a, a' \in N$, then a is definitely not in the starting set. Hence, we just have to focus on individuals a for whom $\varphi(a', a) \neq 0$ holds for each $a' \in N$. Let us denote these individuals by $N^{\neq 0}$. In order to show our result, we define the following flow network. There is a source x, a sink y, a vertex $v(a)$ for each $a \in N$, and a vertex $u(a')$ for each $a' \in N^{\neq 0}$. The edges are defined as follows.

- There is an edge from x to each $v(a)$ with capacity $(n-k)-|0(a,\varphi)|$ ($a \in N$). Notice that a surely disqualifies $|0(a,\varphi)|$ individuals in the partial profile φ. Moreover, a disqualifies exactly $n-k$ individuals in total for each k-extension ϕ. Hence, the capacity indicates how many further individuals may still be disqualified by a so that exactly $n-k$ individuals are disqualified by a in total. For all these individuals a' it holds $\varphi(a,a') = *$.

- There exists an edge from $v(a)$ to $u(a')$ with capacity 1 if and only if $\varphi(a,a') = *$ ($a \in N, a' \in N^{\neq 0}$). These edges ensure that a can assign the remaining [5] disqualifications only to those individuals a' for which $\varphi(a,a') = *$ holds. All other individuals are either definitely qualified or definitely disqualified by a.

- For every $a \in N^{\neq 0}$, there is an edge from $u(a)$ to y with capacity 1.

Analogously to the proof of Lemma 8.4, it follows that the flow network yields an integral flow of size $|N^{\neq 0}|$ (that is, a maximum flow) if and only if there is a k-extension ϕ of φ for which each individual is disqualified by at least one individual in N. In contrast to the other proof, our flow network assigns the further (and, in general, indefinite) disqualifications instead of the (generally) uncertain qualifications. Each individual can assign exactly $(n-k)-|0(a,\varphi)|$ further disqualifications to individuals in N which is expressed by the capacities of the edges starting from the source. The rightmost edges describe that each individual in $N^{\neq 0}$ requires at least one disqualification.

Note that we reject in case we are able to compute a maximum flow. The preprocessing detecting all individuals with certain disqualifications and determining $N^{\neq 0}$ can be done in $O(n^2)$ time. Moreover, as there are $O(n)$ vertices and $O(n^2)$ edges in the flow network, we can determine a maximum integral flow in $O(n^2) \cdot O(n) = O(n^3)$ time due to [136].

Henceforth, we suppose that such a flow does not exist, i.e., each k-extension ϕ yields a non-empty starting set. For each $b \in S$, we regard a subinstance $I(b)$ for which we try to make b socially disqualified for at least one k-extension. According to this, our overall instance is a NO instance if and only if $I(b)$ is a YES subinstance for at least one $b \in S$.

We provide an algorithm based on the observation that an individual qualifying b in a k-extension ϕ of φ must not be socially qualified either, in order to reach a YES subinstance. And these individuals in turn must not be qualified by yet another socially qualified individual, and so on. We thus compute a sequence of sets $(F_i^b)_{i \in \mathbb{N}_0}$ containing all "forbidden" individuals who would—in case of being socially qualified—make b socially qualified, too. In each step we set $H_i^b := N \setminus F_i^b$. In particular, we have $F_0^b = \{b\}$ and $H_0^b = N \setminus \{b\}$. (For the sake of simplicity, we drop the index b in the following.) These sets are computed as follows.

- $F_0 := \{b\}$. We wish that b is not socially qualified. Note that it does not matter whether b qualifies himself [6], as—if possible—every individual in the (non-empty) starting set of at least one k-extension ϕ disqualifies b and so b is (if possible) no member of the starting set.

[5] Intuitively, these further disqualifications are possible but not definite disqualifications. In case $|0(a,\varphi)| + |*(a,\varphi)| = n-k$ or $|1(a,\varphi)| + |*(a,\varphi)| = k$ hold, there is actually no incompleteness concerning a's opinion about individuals in N. These somewhat degenerated cases are explicitly allowed the more so as such cases can occur in various algorithms throughout this chapter, e.g., when some other potential qualification is fixed to one or zero, and the initially incomplete evaluation of a concerning all individuals in N becomes complete during the running of an algorithm.

[6] Note that for $n = 1$ and $k = 1$, b is socially qualified if and only if b qualifies himself. However, our algorithm is based on $n > k \geq 2$.

- For $i \geq 1$, we compute $F_i := F_{i-1} \cup \{a \in N \setminus F_{i-1} : [\exists a' \in F_{i-1} : \varphi(a,a') = 1] \vee [|*(a,\varphi) \cap H_{i-1}| < k - |1(a,\varphi)|]\}$. In other words, we make two classes of individuals join the set of "forbidden" individuals. On the one hand, if a definitely qualifies an individual in F_{i-1}, we let $a \in F_i$ as in case a is socially qualified, a qualifies an individual in F_{i-1}, that individual in turn is either b or qualifies an individual in F_h with $h < i-1$, and so on. Such a sequence of individuals exist—as a mere consequence of the definition of the sets F_i— regardless of how we extend φ. Anyway, it follows that then b is necessarily socially qualified as well provided that an individual in the current set F_i is socially qualified for a given k-extension ϕ. [7] On the other hand, a might not definitely qualify any individual in F_{i-1}, but in each k-extension ϕ there are exactly $k - |1(a,\varphi)|$ individuals a' with $\varphi(a,a') = *$ and $\phi(a,a') = 1$ (otherwise, we do not obtain a k-extension). Note that if there are fewer than $k - |1(a,\varphi)|$ individuals a' in $H_{i-1}(= N \setminus F_{i-1})$ for whom $\varphi(a,a') = *$ holds, a necessarily qualifies at least one individual in F_{i-1} for every k-extension ϕ of φ. Hence, in order to prevent b from being necessarily socially qualified, a must not be socially qualified either.

  We point out that the special case $i = 1$ can be simplified by calculating $F_1 := \{b\} \cup \{a \in N \setminus \{b\} : \varphi(a,b) = 1 \vee [\varphi(a,b) = * \wedge |1(a,\varphi)| + |*(a,\varphi)| = k]\}$. Equivalently speaking, in order to prevent b from being socially qualified for at least one k-extension ϕ, we must hinder all these individuals $a \neq b$ from being socially qualified who definitely qualify b or for whom $\varphi(a,b) = *$ holds, but the $*$ necessarily turns into a 1, in order that ϕ is a feasible k-extension of φ.

- We repeat the previous step until $F_i = F_{i+1} =: F$ [8] for some $i \in \mathbb{N}_0$. In this case, we stop and set further $H_i =: H$.

Given that the starting set $K_0^C(\phi,N)$ is non-empty for every k-extension ϕ of φ, we claim that $I(b)$ is a YES instance if and only if $H \neq \emptyset$ (and for our overall instance, we thus reject as b is socially disqualified for some k-extension ϕ, being isolated from the starting set individuals and all socially qualified individuals given ϕ).

($\Rightarrow$): First assume for contradiction that b is socially disqualified for some k-extension and $H = \emptyset$. As H is empty, we have $F = N$. According to our assumptions from above, the starting set is non-empty for each k-extension ϕ, i.e., for each ϕ there is at least one individual $\bar{a} \in K_0^C(\phi,N)$. Note that all individuals are in F, including the ones belonging to the starting set $K_0^C(\phi,N)$ for at least one k-extension ϕ. Thus, by definition of the sequence $(F_i)_{i \in \mathbb{N}_0}$, for each ϕ and each individual $\bar{a}$ belonging to the starting set of ϕ there exist (1) a sequence of nonnegative integers $i_1, \ldots, i_r$ (where $r \in \mathbb{N}$) with $i_1 > i_2 > \ldots > i_r = 0$, (2) r pairwise different individuals a_{i_h} ($1 \leq h \leq r$), (3) it holds $a_{i_1} = \bar{a}$, (4) $a_{i_r} = b$, and (5) $\phi(a_{i_h}, a_{i_{h+1}}) = 1$ (and $\varphi(a_{i_h}, a_{i_{h+1}}) \in \{*, 1\}$) for each h, $1 \leq h \leq r-1$. Note that $\bar{a} = b$ is possible (which implies $r = 1$, but then b is trivially socially qualified and we obtain a contradiction). Observe that, by definition of the sets F_i, $i \in \mathbb{N}_0$, such a sequence exists for every ϕ and every starting set individual in ϕ. Otherwise, this starting set individual would

[7] One can illustrate the sets F_i as different hierarchies of forbidden sets. As soon as an individual is part of some F_i, it is necessarily attracted by some F_j with smaller index j (if any) until it finally reaches $F_0 = \{b\}$. Thus, each individual in some F_i is connected with b via a sequence of individuals each of whom qualifying their respective successors.

[8] We point out that we traditionally denote a flow with F. Since there is no way of confusion in this context, we make an exception and henceforth define F as the set of "forbidden individuals".

not belong to F. It follows that b is necessarily socially qualified which is a contradiction to our assumption.

($\Leftarrow$): Suppose that $H \neq \emptyset$. We construct a k-extension ϕ of φ, for which b is socially disqualified, as follows. First of all, it holds $\varphi(a,a') = \phi(a,a')$ whenever $\varphi(a,a') \in \{0,1\}$ $(a,a' \in N)$. The question arises how the other, unsure qualifications have to be set in order to obtain such a k-extension ϕ. Let $a \in H$. According to the definition of the sets F_i, $i \in \mathbb{N}_0$, a does not qualify any individual in F (otherwise, $a \in F$, by definition). Moreover, a definitely qualifies $|1(a,\varphi)|$ individuals in H and it is possible that a qualifies $k - |1(a,\varphi)|$ individuals $a' \in H$ with $a' \in *(a,\varphi)$ in a given k-extension (otherwise, a would belong to F due to the construction of the sets F_i). Notice that a may qualify k individuals in H (including himself) which implies that $|H| \geq k$. Now consider other individuals in H. For these individuals, we may argue analogously. Accordingly, we construct ϕ by letting each individual a in H qualify arbitrary k individuals in H such that exactly $k - |1(a,\varphi)|$ individuals in $*(a,\varphi) \cap H$ are qualified by a and the remaining individuals in $*(a,\varphi) \cap H$ (if any) are disqualified. Moreover a disqualifies all individuals in F.

Observe that no matter how we extend the opinions of the individuals in F (such that each individual in F qualifies k individuals as well), no individual in F is in the starting set and no individual in F is socially qualified (as each individual in H disqualifies all individuals in F). Hence, the pivotal individual b, belonging to F, is socially disqualified in ϕ.

As a combined result, if $H \neq \emptyset$, then H has at least cardinality k and at least one individual in the starting set for some k-extension ϕ of φ belongs to H. Note that we did not make use of the condition that an individual in H belongs to the starting set for at least one k-extension, but we implicitly conclude this by assuming that $H \neq \emptyset$ and that the starting set is non-empty for all k-extensions. Since no individual in F belongs to the starting set in our constructed k-extension, we know that an individual in H must belong to $K_0^C(\phi,N)$.

We point out that not necessarily all individuals in H are socially qualified for a given k-extension ϕ, but for at least one k-extension ϕ the set H contains all socially qualified individuals. Possibly, an individual $a \in H$ can be isolated from all other individuals in H and F in a sense that no other individual in H qualifies a in our constructed k-extension ϕ, but a does not qualify any individual in F either (as otherwise a would belong to F), but a qualifies other individuals in H (at least the individual(s) in the starting set of ϕ). One can say that our algorithm computes a smallest possible set F.

Observe that we require at most n iterations to compute F (in the worst-case, in each iteration exactly one individual joins the current iterated set F_i. In each step, it requires $O(n^2)$ time to check whether an individual in H_i necessarily qualifies an individual in F_{i-1}. Hence, the computation of the sets F and H requires $O(n^3)$ time, for a given pivotal individual $b \in S$.

For the f^{LSR} rule, our proof works similarly. Analogously to the f^{CSR} rule, we compute the sets F_i and H_i. We claim that b is socially disqualified for at least one k-extension ϕ of φ if and only if there is a k-extension ϕ of φ such that each individual in F disqualifies himself.

First assume for contradiction that b is socially disqualified, but for every k-extension ϕ there is an individual in $F \cap K_0^L(\phi,N)$. In this case, b is necessarily socially qualified due to the definition of the sequence $(F_i)_{i\in\mathbb{N}_0}$. This holds because then b is necessarily connected with an individual $a_0 \in F$ in the starting set (for at least one ϕ) by a chain of individuals such that each one in this chain qualifies his respective successor, the first individual in this sequence is a_0, and the last individual is b (cf. the reasoning for the consensus-start respecting rule). Therefore, we obtain a contradiction.

Conversely, assume that there is a k-extension ϕ such that $F \cap K_0^L(\phi,N) = \emptyset$. If $H \cap K_0^L(\phi,N) = \emptyset$, we are done as $K_0^L(\phi,N) = K_1^L(\phi,N) = K_2^L(\phi,N) = \ldots = f^{LSR}(\phi,N) = \emptyset$. If there is some $a_0 \in H \cap K_0^L(\phi,N)$, we know that a_0 can qualify k individuals in H for at least one k-extension. Hence, we construct ϕ by making a_0 qualify himself and $k-1$ other individuals in H. This is possible since otherwise a_0 would belong to F by definition. We argue analogously for all other individuals in H. Each of these individuals can qualify only individuals in H, according to the definition of the sequence $(F_i)_{i\in\mathbb{N}_0}$. It follows analogously to the f^{CSR} rule that no individual in F, including b, is socially qualified in ϕ as it is possible that all of them disqualify themselves and each individual in H disqualifies all individuals in F according to ϕ.

In summary, we only have to compute the sets F_i and H_i, $i \in \mathbb{N}_0$, and check whether for each individual $a \in F$ it holds either $\varphi(a,a) = 0$ or both $\varphi(a,a) = *$ and $|1(a,\varphi)| + |*(a,\varphi)| > k$ (in this case, we accept for our subinstance $I(b)$, $b \in S$). These conditions can be checked in $O(n^2)$ time.

In contrast to the f^{CSR} rule, the question of whether an individual is in the starting set is independent on the other individuals' opinions about this individual.

As a combined result, since there are at most $O(n)$ individuals $b \in S$ to check and each subinstance can be checked in $O(n^3)$ time, it follows that the overall problem is in $O(n^4)$ for all values k and both procedural rules. ❑

The following example illustrates how the algorithm works.

Example 8.13. *Let $f \in \{f^{CSR}, f^{LSR}\}$, $k = 2$, $n = 6$, $N = \{a_1,\ldots,a_6\}$, and the profile φ be defined as follows:*

$$\varphi = \begin{pmatrix} 1 & * & 0 & 0 & * & 0 \\ 1 & * & 0 & * & * & 0 \\ * & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & * & 0 & * & 0 \\ 1 & 0 & 0 & * & * & 0 \\ 1 & * & * & 0 & 0 & 0 \end{pmatrix}.$$

Let further $S = \{a_5\}$.

For both procedural rules, our algorithm computes the following "forbidden sets": $F_0 = \{a_5\}$, $F_1 = \{a_3,a_5\}$ (as a_3 qualifies a_5, all other individuals do not necessarily qualify a_5), $F_2 = \{a_3,a_4,a_5\}$ (for each 2-extension, a_4 qualifies either a_3 or a_5 and hence a_4 must not be socially qualified either; no other individual necessarily qualifies at least one of the individuals a_3 and a_5), and $F_3 = \{a_3,a_4,a_5\} = F_2 =: F$ (since neither a_1, a_2, nor a_6 necessarily qualifies any individual in $\{a_3,a_4,a_5\}$). Setting $H := N \setminus F$, we obtain $H = \{a_1,a_2,a_6\}$. As H is non-empty, we construct the following 2-extension according to which a_5 is socially disqualified, given the f^{CSR} rule (the replaced $*$-entries are marked red):

$$\phi = \begin{pmatrix} 1 & \color{red}{1} & 0 & 0 & \color{red}{0} & 0 \\ 1 & \color{red}{1} & 0 & \color{red}{0} & \color{red}{0} & 0 \\ \color{red}{1} & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & \color{red}{1} & 0 & \color{red}{0} & 0 \\ 1 & 0 & 0 & \color{red}{1} & \color{red}{0} & 0 \\ 1 & \color{red}{1} & \color{red}{0} & 0 & 0 & 0 \end{pmatrix}.$$

Observe that, given the consensus-start respecting rule, $a_6 \in H$ is not socially qualified for the constructed 2-extension according to which b is socially disqualified. a_1 is necessarily in the starting set. Remind that one assumption to compute the sequence $(F_i)_{i\in\mathbb{N}_0}$ at all is that for each 2-extension, at least one individual belongs to the starting set.

Since a_5 is socially disqualified according to ϕ, we accept for our subinstance $I(a_5)$ and reject for our instance of f^{CSR}-2-NQI. For the liberal-start respecting rule, b is socially disqualified as well for our k-extension ϕ since a_3, a_4, and a_5 disqualify themselves and thus $K_0^L(\phi,N)\cap F = \emptyset$ holds.

The following example provides an instance where an individual in F is in the starting set for some k-extensions, but yet the individual in S is socially disqualified for some (different) k-extension.

Example 8.14. *Let $f \in \{f^{LSR}, f^{CSR}\}$, $n = 5$, $N = \{a_1,\dots,a_5\}$, $S = \{a_5\}$, $k = 3$, and φ be defined as follows:*

$$\varphi = \begin{pmatrix} 1 & * & 1 & * & 0 \\ 1 & 1 & * & 0 & 1 \\ 1 & * & 1 & * & 0 \\ 1 & * & * & 1 & 0 \\ 1 & * & * & * & * \end{pmatrix}.$$

Under both procedural rules, we compute $F_0 = \{a_5\}$, $F_1 = \{a_2,a_5\}$, and $F_2 = F_1 = F$. Hence, we obtain $H = N\setminus F = \{a_1,a_3,a_4\}$. For the f^{CSR} rule, we already know—due to $H \neq \emptyset$—that a_5 is socially disqualified for at least one 3-extension. We obtain the following 3-extension:

$$\phi = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

The new entries are marked in red. Observe that individuals in F may arbitrarily assign their unsure qualifications (as they remain "among themselves"), while individuals in H must not qualify individuals in F. Observe that precisely the individuals in H are socially qualified. Notice that, although $a_2 \in F$, there are 3-extensions with socially qualified a_2. This is no contradiction to our algorithm, for we only require that at least one individual, belonging to the starting set for at least one 3-extension, must be in H.

For the f^{LSR} rule, we reject for our subinstance (and therefore accept for our NQI instance) as $a_2 \in F$ definitely qualifies himself. When regarding the same example with $\varphi(a_2,a_2) = *$ instead of $\varphi(a_2,a_2) = 1$, we reset:

$$\varphi = \begin{pmatrix} 1 & * & 1 & * & 0 \\ 1 & 0 & * & 0 & 1 \\ 1 & * & 1 & * & 0 \\ 1 & * & * & 1 & 0 \\ 1 & * & * & * & 0 \end{pmatrix}.$$

as an intermediate step, in order that the individuals in F are not in the starting set of the 3-extension with socially disqualified a_5 we like to construct (the changes in the profile compared to the original partial profile are marked red). Analogously to the f^{CSR} rule, we compute a profile ϕ with socially disqualified a_5 given the f^{LSR} rule as follows:

$$\phi = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

Under ϕ, all individuals in H are socially qualified and all individuals in F are socially disqualified according to the f^{LSR} rule. Hence, we reject in total.

8.3 Conclusion

We have studied PQI and NQI under the framework of group identification. In these problems, we are given a partial profile and a subset S of individuals. The question is whether the individuals in S are socially qualified in at least one extension of the given partial profile (PQI) or in every extension of the given partial profile (NQI). In addition, we have considered k-PQI and k-NQI which differ from PQI and NQI in that in the searched extensions we restrict each individual to qualify exactly k individuals.

We have studied the complexity of these problems for the prevalent consent rules and two procedural rules f^{LSR} and f^{CSR}. More precisely, we have derived both polynomial-time algorithms as well as NP-hardness results. In general, our results reveal that most of these problems are polynomial-time solvable. Moreover, for consent rules, the complexity of our polynomial-time algorithms increases slightly as the consent quotes increase. Furthermore, the consent quotes s and t play different roles in the complexity of these problems. For instance, for consent rules $f^{(s,t)}$ the problem k-PQI can be solved in $O(n^2)$ time if $s = 1$ and $t \leq 2$. For $t = 1$ and $s \geq 2$, the running time of the algorithm for k-PQI increases to $O(n^3)$. When $\min(s,t) \geq 2$, the running time of our algorithm for k-PQI increases to $O(n^{(k+t+2)})$, leaving the algorithm to be polynomial-time decidable only when $k+t$ is a constant. By contrast, for k-NQI the running time of our algorithms is $O(n^2)$ for all consent quotes. We have obtained a complexity dichotomy result for k-PQI for f^{LSR} and f^{CSR} with respect to the values of k. In particular, if $k = 1$, k-PQI for f^{CSR} and f^{LSR} can be solved in $O(n^2)$ time; otherwise, it becomes NP-hard. For k-NQI, we have achieved $O(n^2)$-time algorithms for f^{LSR} and f^{CSR} for $k = 1$, whereas the problem is in $O(n^4)$ for $k \geq 2$ for both procedural rules. The intuition that k-PQI and k-NQI for consent rules are generally easy to solve is that whether an individual a is socially qualified can be independently determined by the number of individuals qualifying or disqualifying a. However, in f^{CSR} and f^{LSR}, if an individual a is socially qualified does not depend only on who qualify a, but also on the connectivity between the individuals qualifying a and the initially socially qualified individuals. Between k-PQI and k-NQI, our results reveal that k-NQI is easier to solve (cf. the results in Chapters 3 and 4 for necessary and possible winner both under partial information and for lot-based voting rules; this is in contrast to necessary bribery which is harder than possible bribery for several problems). The reason is that in k-NQI we need only to determine if there is just

	consent rule $f^{(s,t)}$			f^{CSR}	f^{LSR}
	$s+t=2$	$s+t=3 \vee t=1$	$s+t\geq 4 \wedge t>1$		
k-PQI	$O(n^2)$	$s=1: O(n^2)$ $t=1: O(n^3)$	$O(n^{(k+t+2)})$	$k\geq 2$: NPC $k=1: O(n^2)$	$k\geq 2$: NPC $k=1: O(n^2)$
k-NQI	$O(n^2)$	$O(n^2)$	$O(n^2)$	$k=1$: $O(n^2)$ $k\geq 2: O(n^4)$	$k=1$: $O(n^2)$ $k\geq 2: O(n^4)$

Table 8.1: A summary of our results for k-PQI and k-NQI. n denotes the number of individuals. NPC stands for NP-complete.

one individual $a \in S$ and one k-extension of φ with respect to which a is not socially qualified. We refer to Table 8.1 for a summary of our results.

There remain several open questions. For example, investigating the NP-complete problems studied in this chapter from the parameterized complexity point of view would be an interesting research topic. A natural parameter would be $|S|$. As pointed out above, k-PQI for consent rules $f^{(s,t)}$ is fixed-parameter tractable (FPT) with respect to $|S|$. In fact, we would have a single-exponential time algorithm for the problem with respect to $|S|$ (see the discussion after Theorem 8.7). However, whether k-PQI for the two procedural rules f^{CSR} and f^{LSR} is FPT with respect to $|S|$, remains open.

As we have already done in Chapters 3 and 4, one could also study a k-Veto variant in group identification, that is, a setting where each individual disqualifies exactly k individuals.

We further refer to other kinds of incomplete profiles for future research. So individuals might additionally declare partial orders over all individuals and each individual qualified by an individual a must be preferred to all individuals disqualified by a. Or each individual specifies a complete or partial order over N, and for each individual the number of qualifications assigned by this individual is unknown or lies in a certain range. Moreover, it seems to be an appealing idea to regard partial profiles in bribery or control as we have done in Chapter 3 in the context of voting.

Besides, one could define and regard other social rules. One could also consider more general profiles φ in a sense that each individual has more than two options to evaluate the other individuals. So individuals might qualify or disqualify other individuals, or be just indifferent towards another individual. Other, yet more general profiles are thinkable as well.

Going beyond studying the complexity, studying group identification under partial information from the axiomatic perspective (cf. the related work about group identification) appears to be a fascinating direction for future research.

Bibliography

[1] R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, 1993.

[2] G. Ayllón and D. Caramuta. Single-dipped preferences with satiation: strong group strategy-proofness and unanimity. *Social Choice and Welfare*, 47(2):245–264, 2016.

[3] H. Aziz, M. Brill, F. A. Fischer, P. Harrenstein, J. Lang, and H. G. Seedig. Possible and necessary winners of partial tournaments. *Journal of Artificial Intelligence Research*, 54:493–534, 2015.

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

[5] H. Aziz, S. Gaspers, N. Mattei, N. Narodytska, and T. Walsh. Ties matter: Complexity of manipulation when tie-breaking with a random vote. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pages 74–80. AAAI Press, 2013.

[6] H. Aziz, I. Schlotter, and T. Walsh. Control of fair division. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 67–73. IJCAI/AAAI Press, 2016.

[7] Y. Bachrach, N. Betzler, and P. Faliszewski. Probabilistic possible winner determination. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 697–702. AAAI Press, 2010.

[8] M. Balinski and R. Laraki. *Majority Judgment: Measuring, Ranking, and Electing*. MIT Press, Cambridge, MA, 2011.

[9] M. Ball, T. Magnanti, B. Monma, and G. Nemhauser. *Network Models (Handbooks in Operations Research and Management Science)*, volume 7. Elsevier Science B.V., 1995.

[10] H.-J. Bandelt. Networks with Condorcet solutions. *European Journal of Operations Research*, 20:314–326, 1985.

[11] J. Bang-Jensen and G. Gutin. *Digraphs - Theory, Algorithms and Applications*. Springer, London, 2009.

[12] N. Barrot, L. Gourvès, J. Lang, J. Monnot, and B. Ries. Possible winners in approval voting. In *Proceedings of the 3rd International Conference on Algorithmic Decision Theory*, pages 57–70. Springer, 2013.

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

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

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

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

[17] D. Baumeister, G. Erdélyi, O. Erdély, and J. Rothe. Complexity of manipulation and bribery in judgment aggregation for uniform premise-based quota rules. *Mathematical Social Sciences*, 76:19–30, 2015.

[18] D. Baumeister, P. Faliszewski, J. Lang, and J. Rothe. Campaigns for lazy voters: Truncated ballots. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pages 577–584. IFAAMAS, 2012.

[19] D. Baumeister, M. Roos, and J. Rothe. Computational complexity of two variants of the possible winner problem. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 853–860. IFAAMAS, 2011.

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

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

[22] N. Betzler, J. Guo, and R. Niedermeier. Parameterized computational complexity of Dodgson and Young elections. *Information and Computation*, 208(2):165–177, 2010.

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

[24] N. Betzler and J. Uhlmann. Parameterized complexity of candidate control in elections and related digraph problems. *Theoretical Computer Science*, 410(52):5425–5442, 2009.

[25] D. Binkele-Raible, G. Erdélyi, H. Fernau, J. Goldsmith, N. Mattei, and J. Rothe. The complexity of probabilitic lobbying. *Discrete Optimization*, 11:1–21, 2004.

[26] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: A tool for representing and reasoning with conditional ceteris paribus statements. *Journal of Artificial Intelligence Research*, 21:135–191, 2004.

[27] S. Brams and P. Fishburn. Approval voting. *American Political Science Review*, 72(3):831–847, 1978.

[28] F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. Procaccia. *Handbook of Computational Social Choice*. Cambridge University Press, 2015.

[29] R. Bredereck, J. Chen, and G. Woeginger. A characterization of the single-crossing domain. *Social Choice and Welfare*, 41(4):989–998, 2013.

[30] D. Briskorn, G. Erdélyi, and C. Reger. Bribery in k-Approval and k-Veto under partial information (Extended Abstract). In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems*, pages 1299–1300. IFAAMAS, 2016.

[31] L. Chen, L. Xu, S. Xu, Z. Gao, N. Shah, Y. Lu, and W. Shi. Protecting election from bribery: New approach and computational complexity characterization (Extended Abstract). In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 1894–1896. IFAAMAS, 2018.

[32] 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*, pages 762–767. AAAI Press, 2010.

[33] Y. Chevaleyre, J. Lang, N. Maudet, J. Monnot, and L. Xia. New candidates welcome! Possible winners with respect to the addition of new candidates. *Mathematical Social Sciences*, 64(1):74–88, 2012.

[34] W. Cho and B. Ju. Multinary group identification. *Theoretical Economics*, 12:513–531, 2017.

[35] R. Christian, M. Fellows, F. Rosamond, and A. Slinko. On complexity of lobbying in multiple referenda. *Review of Economic Design*, 11(3):217–224, 2007.

[36] T. Coleman and V. Teague. On the complexity of manipulating elections. In *Proceedings of the 13th Conference on Computing: the Australasian Theory Symposium*, pages 25–33. Australian Computer Society, 2007.

[37] V. Conitzer and T. Sandholm. Universal voting protocol tweaks to make manipulation hard. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 781–788. Morgan Kaufmann, 2003.

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

[39] V. Conitzer, T. Walsh, and L. Xia. Dominating manipulations in voting with partial information. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pages 638–643. AAAI Press, 2011.

[40] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1966.

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

[42] P. Dey. Manipulative elicitation - A new attack on elections with incomplete preferences. *Theoretical Computer Science*, 731:36–49, 2018.

[43] P. Dey and N. Misra. On the exact amount of missing information that makes finding possible winners hard. In *Proceedings of the 42nd International Symposium on Mathematical Foundations of Computer Science*, pages 57:1–57:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[44] P. Dey, N. Misra, and Y. Narahari. Kernelization complexity of possible winner and coalitional manipulation problem in voting. *Theoretical Computer Science*, 616:111–125, 2016.

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

[46] P. Dey, N. Misra, and Y. Narahari. Complexity of manipulation with partial information in voting. *Theoretical Computer Science*, 726:78–99, 2018.

[47] D. Dimitrov. The social choice approach to group identification. *Consensual Processes*, pages 123–134, 2011.

[48] D. Dimitrov, S. C. Sung, and Y. Xu. Procedural group identification. *Mathematical Social Science*, 54(2):137–146, 2007.

[49] B. Dorn and D. Krüger. Being caught between a rock and a hard place in an election - voter deterrence by deletion of candidates. In *Proceedings of the 39th International Conference on Current Trends in Theory and Practice of Computer Science*, pages 182–193. Springer, 2013.

[50] B. Dorn and D. Krüger. On the hardness of bribery variants in voting with CP-nets. *Annals of Mathematics and Artificial Intelligence*, 77(3-4):251–279, 2016.

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

[52] E. Elkind and G. Erdélyi. Manipulation under voting rule uncertainty. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pages 624–634. IFAAMAS, 2012.

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

[54] E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. In *Proceedings of the 2nd International Symposium on Algorithmic Game Theory*, pages 299–310. Springer, 2009.

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

[56] E. Elkind, J. Lang, and A. Saffidine. Choosing collectively optimal sets of alternatives based on the Condorcet criterion. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 186–191. IJCAI/AAAI, 2011.

[57] E. Elkind and H. Lipmaa. Hybrid voting protocols and hardness of manipulation. In *16th International Symposium on Algorithms and Computation*, pages 206–215. Springer *Lecture Notes in Computer Science 3827*, 2005.

[58] E. Elkind, D. Pasechnik, and M. Wooldridge. Strategic considerations in the design of committees. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pages 439–446. IFAAMAS, 2013.

[59] P. Emerson. The original Borda count and partial voting. *Social Choice and Welfare*, 40(2):353–358, 2013.

[60] U. Endriss. *Trends In Computational Social Choice*. lulu.com, 2017.

[61] E. Ephrati and J. Rosenschein. The Clarke tax as a consensus mechanism among automated agents. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 173–178. AAAI Press/The MIT Press, 1991.

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

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

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

[65] G. Erdélyi, E. Hemaspaandra, and L. Hemaspaandra. More natural models of electoral control by partition. In *Proceedings of the 4th International Conference on Algorithmic Decision Theory*, pages 396–413. Springer, 2015.

[66] G. Erdélyi, L. Hemaspaandra, J. Rothe, and H. Spakowski. Frequency of correctness versus average polynomial time. *Information Processing Letters*, 109(16):946–949, 2009.

[67] G. Erdélyi, M. Lackner, and A. Pfandler. Computational aspects of nearly single-peaked electorates. *Journal of Artificial Intelligence Research*, 58:297–337, 2017.

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

[69] G. Erdélyi and C. Reger. Possible bribery in k-Approval and k-Veto under partial information. In *Proceedings of the 17th International Conference on Artificial Intelligence: Methodology, Systems, Applications*, pages 299–309. Springer, 2016.

[70] G. Erdélyi and C. Reger. Possible voter control in k-Approval and k-Veto under partial information. In *Proceedings of the 1st Workshop Präferenzen und Personalisierung in der Informatik*, pages 185–192. GI, 2017.

[71] G. Erdélyi, C. Reger, and Y. Yang. The complexity of bribery and control in group identification. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, pages 1142–1150. IFAAMAS, 2017.

[72] G. Erdélyi, C. Reger, and Y. Yang. Complexity of group identification with partial information. In *Proceedings of the 5th International Conference on Algorithmic Decision Theory*, pages 182–196. Springer, 2017.

[73] G. Erdélyi, C. Reger, and Y. Yang. Completing the puzzle: Solving open problems for control in elections. In *Proceedings of the 11th Multidisciplinary Workshop on Advances in Preference Handling*. AAAI Press, 2018.

[74] B. Escoffier, J. Lang, and M. Öztürk. Single-peaked consistency and its complexity. In *Proceedings of the 18th European Conference on Artificial Intelligence*, pages 366–370. IOS Press, 2008.

[75] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 301–312. ACM, 2003.

[76] P. Faliszewski. Nonuniform bribery (Short Paper). In *Proceedings of the 7th Internationa Conference on Autonomous Agents and Multiagent Systems*, volume 3, pages 1569–1572. IFAAMAS, 2008.

[77] P. Faliszewski and E. Elkind. Approximation algorithms for campaign management. In *Proceedings of the 6th Workshop on Internet and Network Economics*, pages 473–482. Springer, 2010.

[78] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. The complexity of bribery in elections. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*, pages 641–646. AAAI Press, 2006.

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

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

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

[82] P. Faliszewski, E. Hemaspaandra, and L. A. Hemaspaandra. Weighted electoral control. *Journal of Artificial Intelligence Research*, 52:507–542, 2015.

[83] P. Faliszewski, P. Slinko, and N. Talmon. The complexity of multiwinner voting rules with variable number of winners. Technical report, 2017.

[84] P. Fishburn. An analysis of simple voting systems for electing committees. *SIAM Journal on Applied Mathematics*, 41(3):499–502, 1981.

[85] Z. Fitzsimmons and E. Hemaspaandra. Complexity of manipulative actions when voting with ties. In *Proceedings of the 4th Inrernational Conference on Algorithmic Decision Theory*, pages 103–119, 2015.

[86] Z. Fitzsimmons and E. Hemaspaandra. The complexity of succinct elections (Extended Abstract). In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 4921–4922. AAAI Press, 2017.

[87] Z. Fitzsimmons, E. Hemaspaandra, and L. Hemaspaandra. Control in the presence of manipulators: Cooperative and competitive cases. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 113–119. IJCAI, 2013.

[88] H. N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of the 15th annual ACM Symposium on Theory of Computing*, pages 448–456. ACM, 1983.

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

[90] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976.

[91] S. Gaspers, V. Naroditskyi, N. Narodytska, and T. Walsh. Possible and necessary winner problem in social polls. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems*, pages 613–620. IFAAMAS, 2014.

[92] W. Gehrlein. The Condorcet criterion and committee selection. *Mathematical Social Sciences*, 10(3):199–209, 1985.

[93] S. Ghosh, M. Mundhe, K. Hernandez, and S. Sen. Voting for movies: The anatomy of recommender systems. In *Proceedings of the 3rd Annual Conference on Autonomous Agents*, pages 434–435. ACM Press, 1999.

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

[95] A. Gibbard. Manipulation of schemes that mix voting with chance. *Econometrica*, 45(3):665–681, 1977.

[96] R. Gibbons. *Game Theory for Applied Economists*. Princeton University Press, 1992.

[97] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[98] U. Grandi, A. Loreggia, F. Rossi, and V. A. Saraswat. From sentiment analysis to preference aggregation. In *International Symposium on Artificial Intelligence and Mathematics*, 2014.

[99] U. Grandi, J. Stewart, and P. Turrini. The complexity of bribery in network-based rating systems. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI Press, 2018.

[100] J. Green-Armytage. Strategic voting and nomination. *Social Choice and Welfare*, 42(1):141–148, 2016.

[101] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, Heidelberg, 1988.

[102] P. Hall. On representation of subsets. *Quarterly Journal of Mathematics (Oxford)*, 10:26–30, 1935.

[103] N. Hazon, Y. Aumann, S. Kraus, and M. Wooldridge. On the evaluation of election outcomes under uncertainty. *Artificial Intelligence*, 189:1–18, 2012.

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

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

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

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

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

[109] M. Hojati. Optimal political districting. *Computers & Operations Research*, 23(12):1147–1161, 1996.

[110] A. Kasher. Jewish collective identity. In *Jewish Identity*, chapter 4, pages 56–78. Temple University Press, U.S., 1993.

[111] A. Kasher and A. Rubinstein. On the question "Who is a J?": A social choice approach. *Logique Analyse*, 160:385–395, 1997.

[112] O. Keller, A. Hassidim, and N. Hazon. Approximating bribery in scoring rules. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI Press, 2018.

[113] D. Kilgour. Approval elections with a variable number of winners. *Theory and Decision*, 81(2):199–211, 2016.

[114] D. Kilgour, S. Brams, and R. Sanver. How to elect a representative committee using approval balloting. In *Mathematics and Democracy: Voting Systems and Collective Choice*, pages 83–95. Springer, 2006.

[115] B. Klaus. Coalitional strategy-proofness in economies with single-dipped preferences and the assignment of an indivisible object. *Games and Economic Behavior*, 34(1):64–82, 2001.

[116] V. Klee and G. Minty. How good is the simplex algorithm. *Inequalities III*, pages 159–175, 1972.

[117] K. Konczak and J. Lang. Voting procedures with incomplete preferences. In *Proceedings of the 3rd IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling*, pages 124–129. IJCAI, 2005.

[118] J. Lang, N. Maudet, and M. Polukarov. New results on equilibria in strategic candidacy. In *6th International Symposium on Algorithmic Game Theory*, pages 13–25. Springer, 2013.

[119] A. Lin. The complexity of manipulating k-Approval elections. In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence*, volume 2, pages 212–218. SciTePress, 2011.

[120] A. Lin. *Solving hard problems in election systems*. PhD thesis, Rochester Institute of Technology, 2012.

[121] H. Liu, H. Feng, D. Zhu, and J. Luan. Parameterized computational complexity of control problems in voting systems. *Theoretical Computer Science*, 410(27-29):2746–2753, 2009.

[122] A. Loreggia. *Iterative Voting, Control and Sentiment Analysis*. PhD thesis, University of Padova, 2016.

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

[124] N. Mattei, M. S. Pini, F. Rossi, and K. B. Venable. Bribery in voting with CP-nets. *Annals of Mathematics and Artificial Intelligence*, 68(1-3):135–160, 2013.

[125] C. Maushagen, M. Neveling, J. Rothe, and A. Selker. Complexity of shift bribery in Hare, Coombs, Baldwin, and Nanson elections. In *Proceedings of the 15th International Symposium on Artificial Intelligence and Mathematics*, 2018.

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

[127] V. Menon and K. Larson. Computational aspects of strategic behaviour in elections with top-truncated ballots. *Autonomous Agents and Multi-Agent Systems*, 31(6):1506–1547, 2017.

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

[129] T. Miasko and P. Faliszewski. The complexity of priced control in elections. *Annals of Mathematics and Artificial Intelligence*, 77(3-4):225–250, 2016.

[130] A. D. Miller. Group identification. *Games and Economic Behavior*, 63(1):188–202, 2008.

[131] N. R. Miller. Monotonicity failure under STV and related voting systems. Technical report, 2002.

[132] N. Narodytska and T. Walsh. The computational impact of partial votes on strategic voting. In *Proceedings of the 21st European Conference on Artificial Intelligence*, pages 657–662. IOS Press, 2014.

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

[134] H. Nicolas. "I want to be a J!": Liberalism in group identification problems. *Mathematical Social Sciences*, 54(1):59–70, 2007.

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

[136] J. B. Orlin. Max flows in O(nm) time, or better. In *Proceedings of the 45th ACM Symposium on the Theory of Computing Conference*, pages 765–774. ACM, 2013.

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

[138] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Incompleteness and incomparability in preference aggregation: Complexity results. *Artificial Intelligence*, 175(7-8):1272–1289, 2011.

[139] A. Procaccia and J. Rosenschein. The distortion of cardinal preferences in voting. In *Proceedings of the 10th International Workshop on Cooperative Information Agents X*, pages 317–331. Springer, 2006.

[140] A. Procaccia and J. Rosenschein. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research*, 28:157–181, 2007.

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

[142] T. Ratliff. Some startling inconsistencies when electing committees. *Social Choice and Welfare*, 21(3):433–454, 2003.

[143] C. Reger. Voter control in k-Approval and k-Veto under partial information. In *Proceedings of the 14th International Symposium on Artificial Intelligence and Mathematics*, 2016.

[144] A. Rey, J. Rothe, and A. Marple. Path-disruption games: Bribery and a probabilistic model. *Theory of Computing Systems*, 60(2):222–252, 2017.

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

[146] J. Rothe. *Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*. Springer, 2016.

[147] D. Samet and D. Schmeidler. Between liberalism and democracy. *Journal of Economic Theory*, 110(2):213–233, 2003.

[148] S. Sato. Informational requirements of social choice rules to avoide the Condorcet loser: A characterization of the plurality with a runoff. *Mathematical Social Sciences*, 79:11–19, 2016.

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

[150] I. Schlotter, P. Faliszewski, and E. Elkind. Campaign management under approval-driven voting rules. *Algorithmica*, 77(1):84–115, 2017.

[151] A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003.

[152] E. Shimon. An algorithm for determining whether the connectivity of a graph is at least k. *SIAM Journal on Computing*, pages 4(3):393–396, 1975.

[153] H. Steinhaus. The problem of fair division. *Econometrica*, 16(1):101–104, 1948.

[154] B. Sziklai. How to identify experts in a community. *International Journal of Game Theory*, 47(1):155–173, 2018.

[155] E. Szpilrajn. Sur l'extension de l'ordre partiel. *Fundamenta Mathematicae*, 16:386–389, 1930.

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

[157] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[158] E. van Damme. *Stability and Perfection of Nash Equilibria*, volume 2. Springer, 1991.

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

[160] T. Walsh and L. Xia. Lot-based voting rules. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, volume 3, pages 603–610. IFAAMAS, 2012.

[161] J. Wang, W. Su, M. Yang, J. Guo, Q. Feng, F. Shi, and J. Chen. Parameterized complexity of control and bribery for d-approval elections. *Theoretical Computer Science*, 595:82–91, 2015.

[162] D. B. West. *Introduction to Graph Theory*. Prentice-Hall, 2000.

[163] B. Wilder and Y. Vorobeychik. Controlling elections through social influence. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 265–273. IFAAMAS, 2018.

[164] K. Wojtas and P. Faliszewski. Possible winners in noisy elections. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 1499–1505. AAAI Press, 2012.

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

[166] L. Xia. Designing social choice mechanisms using machine learning. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, pages 471–474. IFAAMAS, 2013.

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

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

[169] L. Xia, J. Lang, and J. Monnot. Possible winners when new alternatives join: new results coming up! In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 829–836. IFAAMAS, 2011.

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

[171] Y. Yang and D. Dimitrov. How hard is it to control a group? *Autonomous Agents and Multi-Agent Systems*, 32:1–21, 2018.

[172] Y. Yang and J. Guo. How hard is control in multi-peaked elections: A parameterized study (Extended Abstract). In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pages 1729–1730. IFAAMAS, 2015.

[173] Y. Yang and J. Guo. Possible winner problems on partial tournaments: A parameterized study. *Journal of Combinatorial Optimization*, 33(3):882–896, 2017.

[174] Y. Yang, Y. Shresta, and J. Guo. How hard is bribery in party based elections (Extended Abstract). In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pages 1725–1726. IFAAMAS, 2015.

[175] Y. Yang and J. Wang. Anyone but them: The complexity challenge for a resolute election controller. In *Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems*, pages 1133–1141. IFAAMAS, 2017.

Appendix

Transformation Algorithm to obtain a R3DM instance

SIMPLIFICATION RULE FOR (≤ 3)-3DM	
Given:	An instance (X,Y,Z,E) of (≤ 3)-3DM with $\|X\| = \|Y\| = \|Z\| = n$, $\|E\| = m$, and $l(a) \leq 3$ for each $a \in X \cup Y \cup Z$, where $l(a)$ denotes the number of triples containing $a \in X \cup Y \cup Z$. Let further $C_i := \{a \in X \cup Y \cup Z : \ l(a) = i\}$ $(0 \leq i \leq 3)$.
Goal:	Transform (if possible) our instance into an instance (X',Y',Z',E') of R3DM, i.e., it holds $l(a) \in \{2,3\}$ for each $a \in X' \cup Y' \cup Z'$.

Let E' be a collection of triples which are fixed while running the simplification algorithm defined in the following. Our algorithm is based on the observation that in case an element $a \in X \cup Y \cup Z$ occurs in exactly one triple, this triple must belong to a three-dimensional matching, provided that one exists. The initially empty matching E' gets larger and larger until we (1) find a three-dimensional matching, (2) notice that a matching is impossible, or (3) obtain a smaller instance of our original problem for which every element occurs in two or three triples. Our simplication algorithm does as follows:

Initialization: $E' = \emptyset$, C_0, C_1, C_2, C_3, and E (the latter five sets correspond to the original instance). Go to step 1.

1. If $C_0 \neq \emptyset$, output NO as a matching is impossible. Our algorithm halts. Otherwise, go to step 2.

2. If $C_0 = \emptyset$, $C_1 \neq \emptyset$, select an arbitrary $a \in C_1$. Fix $e = \{x^e, y^e, z^e\}$ with $e \ni a$ in the matching, that is, $E' = E' \cup \{e\}$ (observe that $a = x^e$ or $a = y^e$ or $a = z^e$ holds). [1] As a feasible matching must not contain any further triple with elements in e, delete all triples e' with $e \cap e' \neq \emptyset$ from E. Update our instance as follows:

$$E = E \setminus \{e' \in E : \ e' \cap e \neq \emptyset\}, \quad X \cup Y \cup Z = (X \cup Y \cup Z) \setminus \{x^e, y^e, z^e\}.$$

[1] Although this is not entirely correct, we regard e as a three-element subset of $X \cup Y \cup Z$ including one element from X, Y, and Z each.

Observe that all remaining elements are not yet assigned to any triple in the current matching. Compute C_i for our residual instance. If $|C_0 \cup C_1| > 0$, go back to step 1. Otherwise, go to step 3.

3. If $C_0 = C_1 = \emptyset$ and $C_2 \cup C_3 \neq \emptyset$, each element belongs either to C_2 or C_3, and we arrive at an instance of R3DM. Our algorithm halts in this case. Otherwise, go to step 4.

4. If $|E'| = n$ (and thus $C_0 = C_1 = C_2 = C_3 = \emptyset$), our matching is, by construction, a three-dimensional matching (as our construction ensures that no element occurs in two or more triples in E', and $|E'| = n$ is thus only possible when all elements occur in exactly one triple). Hence, our algorithm accepts and halts.

Observe that our algorithm either accepts our original instance, rejects this instance, or reduces the original (≤ 3)-3DM instance to a R3DM instance. Note that it is possible that our initial instance is already a R3DM-instance. In this case, our algorithm immediately jumps to step 3 and comes to a halt without accepting or rejecting our problem. Interestingly, this algorithm can be modeled via a DTM that halts for every input, but for some inputs (namely, when we actually compute a R3DM-instance) it neither accepts nor rejects our original (≤ 3)-3DM instance.

We point out that this algorithm is probably existing in literature, but we have not found it. Thus, we have provided this algorithm just in case.

Proof Sketch of Theorem 3.8

Claim: k-APPROVAL-X-POSSIBLE WINNER and k-VETO-X-POSSIBLE WINNER are in P for each $k \in \mathbb{N}$ and each model $X \in \{\text{Gaps}, \text{FP}, \text{1Gap}, \text{BTO}, \text{TTO}, \text{CEV}\}$.

Proof Sketch. Let (C,V) be an election with $m > k$ candidates in candidate set $C = \{c_1, \ldots, c_{m-1}, c\}$, n voters in voter set $V = \{v_1, \ldots, v_n\}$ (w.l.o.g. $n \in \mathbb{N}$) all according to the same model $X \in \{\text{FP}, \text{Gaps}\}$ (all other models are special cases of both of them), and a designated candidate $c \in C$.

First regard k-Approval. As a first step, we determine in each vote v the candidates definitely approved (C_A^v), possibly but not definitely approved ($C_?^v$), and definitely disapproved by v (C_D^v). To do so, we apply Theorem 3.5. If $c \in C_?^v$, we reset $C_A^v = C_A^v \cup \{c\}$ and $C_?^v = C_?^v \setminus \{c\}$. Possibly, $C_?^v$ becomes singleton after that and the vote artificially becomes complete, by fixing c on an approval position and the remaining candidate in $C_?^v$ on a disapproval position.

After this preprocessing, we compute the potential score $pscore(c)$ embracing all definite and unsure approvals for c. Moreover, $score(c_j)$ denotes the definite approval score of c_j ($1 \leq j \leq m-1$). If $score(c_j) > pscore(c)$ for some c_j, we immediately reject as c_j beats c for every extension. Otherwise, we check whether c is a winner for at least one completion by trying to distribute the unsure approvals to non-distinguished candidates in a way that no other candidate has more points than c. Let us assume that voter v_i, $1 \leq i \leq n$, assigns α_i unsure approvals to non-distinguished candidates (unsure approvals for c are already fixed as approvals) which in turn means that $k - \alpha_i$ approvals assigned by v_i are already fixed. We define the following b-edge matching problem for the bipartite and simple graph G, defined by the vertices $\mathscr{V}$ and edges $\mathscr{E}$ defined next. We let $\mathscr{V} = V \cup (C \setminus \{c\})$. The edges in $\mathscr{E}$ are defined as follows. There is an edge $\{v_i, c_j\}$ if and only if c_j is possibly but not definitely approved by v_i ($1 \leq i \leq n$, $1 \leq j \leq m-1$). The upper capacity constraints are $b(v_i) = \alpha_i$ (this number of unsure approvals have to be assigned) and $b(c_j) = pscore(c) - score(c_j)$ (c_j may still receive this number of uncertain approvals; if c_j gets more unsure approvals, c is beaten by c_j). Note that $b(c_j) \geq 0$ due to our assumption.

We claim that a maximum matching with $\sum_{i=1}^{n} \alpha_i$ edges exists if and only if c is a possible winner.

In case this matching exists, we extend our partial profile by letting each voter v_i approve of c_j if the edge $\{v_i, c_j\}$ is in the matching. All definite approvals are already fixed. Due to the size of the matching, all unsure approvals are considered. Since all constraints $b(c_j)$ are satisfied, no c_j has more points than c.

Conversely, in case the maximum matching has smaller size, selecting exactly $\sum_{i=1}^{n} \alpha_i$ edges (one-to-one corresponding to the unsure approvals assigned by the voters in V) implies that some c_j beats c (otherwise, the matching would not be maximum). Hence, c cannot be a possible winner. Note that a feasible matching always exists as the empty matching is always feasible given our assumption $pscore(c) \geq score(c_j)$ for each c_j, $1 \leq j \leq m-1$.

For k-Veto, we may apply the same algorithm by regarding $(m-k)$-Approval. Note that the reduction to the matching remains polynomial.

Observe that this proof is a mere adaption of the proof by Baumeister et al. [18]. Both proofs are modeled as assignment problems where a certain number of unknown approval positions are assigned to candidates. In contrast to this proof sketch, their proof works with flow maximization.

Proof Sketch of Theorem 3.11

Let $\tilde{C} \subseteq C$ denote the subset completely ranked by each voter according to the 1TOS structure.

Claim: 3-VETO-1TOS-POSSIBLE WINNER (under the co-winner model) is in P under the case where $c \in \tilde{C}$ and $C \setminus \tilde{C} = \{p,q\}$.

Proof Sketch. We are given an election (C,V) with m candidates in candidate set C, n voters in voter set V, and a designated candidate $c \in C$. Each voter completely ranks the candidates in $\tilde{C} \subseteq C$, where $c \in \tilde{C}$ and $C \setminus \tilde{C} = \{p,q\}$, $p \neq q$. W.l.o.g., we let $n > 0$. Moreover, we restrict ourselves to $|C| \geq 5$. Otherwise, our voting rule is either the constant scoring rule (when $|C| \leq 3$) or the Plurality rule (for $|C| = 4$) for which our problem is easy even for the more general model PC [21].

Notice that last positions in subelection $(\tilde{C},V)$ correspond to definite vetoes, while each voter possibly, but not definitely vetoes p, q, and his second and third least preferred candidates in $\tilde{C}$. All other candidates are surely not vetoed by the voter. Finally, if a voter vetoes his third to least preferred candidate in $\tilde{C}$ in a given extension, the second to least preferred candidate in $\tilde{C}$ must be vetoed, too (observe that the voters' least preferred $\tilde{C}$ candidates are definitely vetoed).

Keeping this in mind, we transform our problem to a generalized b-edge cover problem similar to the one in the proof of Theorem 3.34. Before we do so, we point out that c has $v_1(c)$ definite vetoes (where $v_1(c')$ denotes the number voters ranking $c' \in \tilde{C}$ last among the candidates in $\tilde{C}$; w.l.o.g., unsure vetoes do not count for c). Then p and q must have at least $2v_1(c)$ vetoes in total. Suppose that at least α_2 voters veto both p and q, while at least α_1 voters veto either p or q for a given extension. We may restrict ourselves to $(\alpha_1,\alpha_2) \in \mathbb{N}_0^2$ satisfying $2\alpha_2 + \alpha_1 = 2v_1(c)$. Observe that α_1 is even. If p and q have exactly $2v_1(c)$ vetoes altogether, the partial votes can be extended in a way that both candidates have exactly $v_1(c)$ vetoes. We call (α_1,α_2) a *guess*. Based on such a guess, we define the following generalized b-edge cover problem:

Let $G = (\mathscr{V},\mathscr{E})$ be an undirected graph with vertices $\mathscr{V} = \tilde{C} \dot{\cup} K \dot{\cup} \{*\}$, where K is a set of auxiliary vertices. The edges are according to Construction I (defined in the proof of Theorem 3.34) and are defined as follows:

- A voter v voting $\tilde{C} \setminus \{c',c'',c'''\} \succ_v c''' \succ_v c'' \succ_v c'$ yields two vertices k_1^v and k_2^v in K and the four edges $\{c'',k_1^v\}$, $\{k_1^v,k_2^v\}$, $\{k_2^v,c'''\}$, and $\{k_2^v,*\}$.

The capacities are $b_l(c) = b_u(c) = 0$ (unsure vetoes never count for c), $b_l(c') = \max(0, v_1(c) - v_1(c'))$, $c' \in \tilde{C} \setminus \{c\}$ (this capacity displays the additional number of unsure vetoes that must count for c' in order that c' has at least as many vetoes as c), $b_l(*) = \alpha_1$, and $b_l(k_h^v) = b_u(k_h^v) = 1$ ($v \in V$, $h = 1,2$). All remaining upper capacities are unlimited.

Observe that, provided that one exists [2], a cover yields either two edges or one edge for each voter (see below). Hence, any feasible cover has a cardinality between n and $2n$. Moreover, at least α_2 voters must yield one edge $\{k_1^v,k_2^v\}$ as only these votes assign two vetoes for p and q. [3]

[2] The largest cover that might exist has $2n$ edges and no voter yields an edge $\{k_1^v,k_2^v\}$ in the cover. In case the largest possible cover is infeasible, for each extension there is some $c' \in \tilde{C} \setminus \{c\}$ vetoed by fewer than $v_1(c)$ voters.

[3] Possibly, there are considerably more voters vetoing either p or q (that is, the capacity constraint $b_l(*)$ is "overfulfilled" and even with $\alpha_2 - 1$ voters vetoing both p and q in a given extension, p and q have at least $2v_1(c)$ altogether. This case, however, is also captured by a cover based on another guess (α_1,α_2).

Similarly to the arguing the proof of Theorem 3.34 and 3.44, it follows that c is a possible winner if and only if there is a guess (α_1, α_2) and a cover with at most $2n - \alpha_2$ edges according to this guess. Given v, the following edges may occur in a cover (each other possibility leads to a contradiction since k_1^v and k_2^v each occur exactly once in any feasible cover):

1. If $\{k_1^v, k_2^v\}$ is in the cover, voter v vetoes p and q for our extension with co-winner c.
2. If $\{k_1^v, c''\}$ and $\{k_2^v, c'''\}$ belong to the cover, voter v vetoes c'' and c'''.
3. If $\{k_1^v, c''\}$ and $\{k_2^v, *\}$ are in the cover, v vetoes c'' and either p or q.

Note that v definitely vetoes his least preferred candidate in $\tilde{C}$, regardless of how we distribute the other two vetoes according to one of the three listed possibilities.

In case a cover with at most $2n - \alpha_2$ edges exists, there are at least α_2 voters v yielding one edge in the cover and thus p and q have enough vetoes altogether, additionally taking into account that at least $b_l(*)$ edges in the cover are incident to $*$.

Overview of all 4-combinations of approved candidates in Example 3.9, part (c)

Recall that $C = \{a, \ldots, g\}$, $\mathscr{F}$ =4-Approval, X =PC, and a voter v specifies the pairwise comparisons $\Pi^v = \{(a,b),(a,c),(a,d),(b,f),(c,e),(c,g),(d,g)\}$.

Note that it suffices to focus on all twenty 4-combinations with a being approved. There exist 15 other 4-combinations without a, but these ones do not correspond to our four approved candidates for any extension. The table below gives an overview of all 4-combinations of approved candidates. Notice that an extension of v may have exactly the following quadruples of approved candidates (a,b,c,d), (a,b,c,e), (a,b,c,f), (a,b,d,f), (a,c,d,e), and (a,c,d,g). Observe that an approval for g implies an approval for c, d, and a. An approval for f implicates b and a being approved. Likewise, an approval for e implies that c and a are approved. This relatively simple example gives an idea of how complex these interdependencies might become for growing k or m.

Approved Candidates	feasible?	Approved Candidates	feasible?
(a,b,c,d)	✓	(a,c,d,e)	✓
(a,b,c,e)	✓	(a,c,d,f)	$\lightning\ (b \succ_v f)$
(a,b,c,f)	✓	(a,c,d,g)	✓
(a,b,c,g)	$\lightning\ (d \succ_v g)$	(a,c,e,f)	$\lightning\ (b \succ_v f)$
(a,b,d,e)	$\lightning\ (c \succ_v e)$	(a,c,e,g)	$\lightning\ (d \succ_v g)$
(a,b,d,f)	✓	(a,c,f,g)	$\lightning\ (b \succ_v f)$
(a,b,d,g)	$\lightning\ (c \succ_v e)$	(a,d,e,f)	$\lightning\ (c \succ_v e)$
(a,b,e,f)	$\lightning\ (c \succ_v e)$	(a,d,e,g)	$\lightning\ (c \succ_v e)$
(a,b,e,g)	$\lightning\ (c \succ_v e)$	(a,d,f,g)	$\lightning\ (b \succ_v f)$
(a,b,f,g)	$\lightning\ (d \succ_v g)$	(a,e,f,g)	$\lightning\ (b \succ_v f)$

Table A.1: Overview of all feasible 4-combinations of approved candidates for Example 3.9. Key: ✓means that the approval combination is feasible, $\lightning$ means that the considered combination inheres at least one contradiction. One such contradiction is given in brackets behind the lightning symbol. For instance, the combination (a,b,c,g) hurts the condition that g's approval implies an approval for d since v prefers d over g. According to this, each feasible approval combination satisfies the conditions $a \succ_v c'$ for each $c' \in C \setminus \{a\}$, $b \succ_v f$, $c \succ_v e$, $c \succ_v g$, and $d \succ_v g$.

Proof of Lemma 3.24

Lemma 3.24: Let $(\delta(c_1),\ldots,\delta(c_{m-1})) \in \mathbb{N}_0^{m-1}$ (one can interpret $\delta(c_j)$ as the additional number of vetoes c_j still requires from the bribed voters). Then the following holds under the assumption $k < m$:

ℓ voters can feasibly assign $\delta(c_j)$ vetoes to c_j (for each $j = 1,\ldots,m-1$) if and only if it holds

$$\delta(c_j) \leq \ell \quad (1 \leq j \leq m-1) \quad \text{and} \quad \sum_{j=1}^{m-1} \delta(c_j) \leq k\ell.$$

Proof. ($\Rightarrow$): Suppose for contradiction that $\delta(c_j) > \ell$ for some j or $\sum_{j=1}^{m-1} \delta(c_j) > k\ell$. In the first case, some c_j requires more than ℓ additional vetoes. This means that one bribed voter vetoes c_j at least twice which is a contradiction. Suppose for contradiction that the second inequality is hurt, but w.l.o.g. the first inequality holds for each c_j (if the first inequality is hurt as well, we arrive again at the first case). Since ℓ voters assign $k\ell$ vetoes in total, the total demand of additional vetoes for non-distinguished candidates exceeds the number of vetoes that bribed voters can assign to them. Again, there is a contradiction.

($\Leftarrow$): Assume that both conditions hold. Since the total demand of additional vetoes can be covered, we merely have to construct an algorithm that assigns the required vetoes to each c_j and none of them is vetoed at least twice by the same voter. For our purposes, we let A be a matrix with ℓ rows and k columns. We associate rows with voters and columns with vetoes. E.g., a_{ir} denotes the rth veto assigned by the ith voter. One could interpret the first and last column as the best and worst veto position in each vote, respectively. Let us first illustrate the basic idea of our algorithm. It begins in field a_{11} and writes c_1 onto the first $\delta(c_1)$ fields in the first column (that is, the first $\delta(c_1)$ voters assign their first=best veto position to c_1). As many other voters as possible place c_2 onto their best veto position. If $\delta(c_2) > \ell - \delta(c_1)$, the first $\delta(c_2) - (\ell - \delta(c_1))$ voters assign their second best veto position to c_2, and so on. By this, the algorithm runs from the upper left corner of the matrix towards the lower right corner; more precisely, it runs from top to bottom in each column, then moves to the top of the next column on the right, until the first $\sum_{j=1}^{m-1} \delta(c_j)$ veto positions of the matrix are filled.

Formally, our algorithm is defined as follows. Let i be the row index, r the column index, and j the candidate index. Initially, we have $i = 0$, $j = 1$, and $r = 1$. While $j \leq m-1$, do the following:

1. If $i + \delta(c_j) \leq \ell$, reset $a_{lr} = c_j$ for each $i+1 \leq l \leq i+\delta(c_j)$. Update $i = i+\delta(c_j)$, $j = j+1$, r remains unchanged. Repeat step 1. Otherwise, go to step 2.

2. If $i+\delta(c_j) > \ell$, reset $a_{lr} = c_j$ for each $i+1 \leq l \leq \ell$ and $a_{l,r+1} = c_j$ for each $1 \leq l \leq \delta(c_j)+i-\ell$ (it comes to a column break). Update $r = r+1$, $i = \delta(c_j)+i-\ell$, $j = j+1$. Go back to step one.

As soon as $j = m$ holds after increasing j by one, our algorithm halts (either in 1. or 2.).

For $j = m$, greedily fill the open positions in A (if any). In each row i, there is some column ρ such that all entries $a_{i\rho},\ldots,a_{ik}$ are empty and all entries $a_{i1},\ldots,a_{i,\rho-1}$ are filled. W.l.o.g. fill these open positions step by step with the presently lexicographically smallest c_j (according to the index)

not yet being vetoed by the ith voter. This is possible as there are $m-1 \geq k$ non-distinguished candidates according to our assumptions (for $k = m$, all candidates including c would be vetoed).

Observe that our algorithm fills $\sum_{j=1}^{m-1} \delta(c_j) \leq \ell k$ positions after scanning the veto demands for $m-1$ non-distinguished candidates. It remains to show that no candidate is vetoed at least twice by the same voter. If all c_j-vetoes fit in the same column, this trivially holds. Otherwise, there is some $l \in \{2,\ldots,\ell\}$ and some column index $r \in \{1,\ldots,k-1\}$ such that c_j is vetoed by the voters $l,\ldots,\ell$ in column r and by the voters $1,\ldots,\delta(c_j)-(\ell-l+1)$ in column $r+1$. Or, formally, it holds $a_{lr} = a_{l+1,r} = \ldots = a_{\ell r} = c_j$ and $a_{1,r+1} = \ldots, a_{\delta(c_j)-\ell+l-1,r+1} = c_j$. In case some voters assigned two vetoes to c_j, at least voter l would veto c_j twice. Then, however, it would hold $\delta(c_j)-\ell+l-1 \geq l$ which in turn is equivalent to $\delta(c_j) \geq \ell+1$. This is impossible due to $\delta(c_j) \leq \ell$ according to our assumptions. Hence, we obtain a contradiction. ❑

Detailed Calculations for Example 3.35

Example 3.35:

(a) Let (C,V) be an election with candidate set $C = \{c, c', c'', p\}$, voter set $V = \{v_1, \ldots, v_4\}$, distinguished candidate c, and bribery limit $\ell = 1$. Each vote is partial according to the 1TOS model and $\tilde{C} = \{c, c', c''\}$ denotes the totally ordered subset. Finally, the voting rule is $\mathscr{F} = 2$-Approval. The voters vote as follows:

$$v_1 : c' \succ c \succ c'', \quad v_2 : c'' \succ c' \succ c, \quad v_3, v_4 : c' \succ c'' \succ c.$$

(b) Let (C,V) be an election with candidate set $C = \{c, c', c'', c''', p_1, p_2\}$, voter set $V = \{v_1, v_2, v_3\}$, distinguished candidate c, and bribery limit $\ell = 1$. Each vote is partial according to the 1TOS model and $\tilde{C} = \{c, c', c'', c'''\}$ denotes the totally ordered subset. Moreover, the voting rule is $\mathscr{F} = 2$-Approval. The voters' votes are as follows (we restrict the rankings to the voters' two favorite candidates in $\tilde{C}$):

$$v_1, v_2 : c' \succ c'', \quad v_3 : c' \succ c'''.$$

Part (a) yields a graph $G = (\mathscr{V}, \mathscr{E})$ defined as follows. The vertices are given by $\mathscr{V} = \{c', c'', p, b, v_1\} \cup \{k_h^{v_i} : 1 \leq h \leq 2,\ 2 \leq i \leq 4\}$. The edge set is given by $\mathscr{E} =: \mathscr{E}_1 \cup \mathscr{E}_2 \cup \mathscr{E}_3$ which are defined as follows:

- $\mathscr{E}_1 := \{\{v_1, c'\}\}$ (i.e., the edge corresponding to the voter potentially approving of c),
- $\mathscr{E}_2 := \{\{c'', k_1^{v_2}\}, \{k_2^{v_2}, c'\}, \{k_2^{v_2}, p\}, \{c', k_1^{v_3}\}, \{k_2^{v_3}, c''\}, \{k_2^{v_3}, p\}, \{c', k_1^{v_4}\}, \{k_2^{v_4}, c''\}, \{k_2^{v_4}, p\}\} \cup \{\{k_1^{v_i}, k_2^{v_i}\} : 2 \leq i \leq 4\}$ (i.e., the edges according to voters disapproving of c), and
- $\mathscr{E}_3 := \{\{c', b\}, \{c'', b\}, \{p, b\}\}$ (each bribed voter assigns one approval to one candidate other than c after the bribery).

Finally, we obtain the following capacities: $b_u(c') = b_u(c'') = b_u(p) = pscore(c) + \ell = 1 + 1 = 2$, $b_l(b) = b_u(b) = 1$, $b_l(v_1) = b_u(v_1) = 1$, and $b_l(k_h^{v_i}) = b_u(k_h^{v_i}) = 1$ ($1 \leq h \leq 2$, $2 \leq i \leq 4$). All other lower capacities, not listed here, are zero.

We accept if and only if the capacity of the maximum matching is at least $2n - pscore(c) = 8 - 1 = 7$. By setting $M = \{\{v_1, c'\}, \{c'', k_1^{v_2}\}, \{k_2^{v_2}, p\}, \{c', k_1^{v_3}\}, \{k_2^{v_3}, p\}, \{k_1^{v_4}, k_2^{v_4}\}, \{b, c''\}\}$, we obtain a feasible matching of desired size. According to the matching, we obtain the following bribery strategy and extension of the remaining votes:

- Voter v_1 approves of c and c'.
- v_2 approves of c'' and p.
- v_3 approves of c' and p.
- v_4 is bribed and approves of c and c'' after the bribery (due to the edge $\{b, c''\}$).

In (b), we define a graph $G = (\mathscr{V}, \mathscr{E})$ with vertices $\mathscr{V} = \{c', c'', c''', *, b\} \cup \{k_h^{v_i} : 1 \leq h \leq 2,\ 1 \leq i \leq 3\}$ and edges $\mathscr{E} =: \mathscr{E}_2 \cup \mathscr{E}_3$ which are defined as follows:

- $\mathscr{E}_2 := \{\{c', k_1^{v_1}\}, \{k_2^{v_1}, c''\}, \{k_2^{v_1}, *\}, \{c', k_1^{v_2}\}, \{k_2^{v_2}, c''\}, \{k_2^{v_2}, *\}, \{c', k_1^{v_3}\}, \{k_2^{v_3}, c'''\}, \{k_2^{v_3}, *\} \cup \{\{k_1^{v_i}, k_2^{v_i}\} : 1 \leq i \leq 3\}\}$ (edges according to voters disapproving of c in the initial election) and
- $\mathscr{E}_3 := \{\{b, c'\}, \{b, c''\}, \{b, c'''\}, \{b, *\}\}$ (each bribed voter assigns one approval to some candidate other than c after the bribery).

We have $\mathscr{E}_1 = \emptyset$ as no voter approves of c before the bribery. Observe that $pscore(c) = 0$ and $pscore_{(C,\bar{V})}(c) = 1$. Hence, we obtain the capacities $b_u(c') = b_u(c'') = b_u(c''') = 1$, $b_l(b) = b_u(b) = b_l(k_h^{v_i}) = b_u(k_h^{v_i}) = 1$ ($1 \leq h \leq 2$, $1 \leq i \leq 3$), and $b_u(*) = \alpha_1$. All lower capacities not listed here are equal to zero.

Since c has one potential point in the final election, p_1 and p_2 may have at most $|\{p_1, p_2\}| \cdot pscore_{(C,\bar{V})}(c) = 2 \cdot 1 = 2$ points in total. We check the two guesses $\alpha := (\alpha_1, \alpha_2) \in \{(2,0), (0,1)\}$ (where $2\alpha_2 + \alpha_1 = 2$, $\alpha_1, \alpha_2 \in \mathbb{N}_0$). We accept if and only if for at least one guess α, there is a matching with at least $2n - pscore(c) - \alpha_2 = 2 \cdot 3 - 0 - \alpha_2 = 6 - \alpha_2$ edges.

- $\alpha = (0,1)$. We search for at most one voter to be left unchanged and approving of both p_1 and p_2. Due to $b_u(*) = 0$, there must not be any voter approving of either p_1 or p_2. A feasible matching of cardinality $5 = 6 - \alpha_2$ is given by $M = \{\{c', k_1^{v_1}\}, \{k_2^{v_1}, c''\}, \{k_1^{v_2}, k_2^{v_2}\}, \{k_1^{v_3}, k_2^{v_3}\}, \{b, c'''\}\}$. Bribing and completing the votes according to M makes c a winner for the constructed extension: v_1 is left unchanged and approves of c' and c''. Voter v_2 is left unchanged and approves of p_1 and p_2. Moreover, voter v_3 is bribed and approves of c and c''' after the bribery. Hence, each candidate has one point for this extension and c is a winner. Note that v_2 and v_3 could change parts. In general, whenever there are $\ell + \alpha_2$ voters each of whom corresponds to one edge $\{k_1^{v_i}, k_2^{v_i}\}$ in the matching, any ℓ voters among them may be bribed and the other α_2 ones remain unchanged and approve of two candidates in $C \setminus \tilde{C}$. Or, even more generally, as many as up to ℓ such voters v_i yielding one edge $\{k_1^{v_i}, k_2^{v_i}\}$ in the matching are bribed and the remaining ones, if any, are left unchanged by the briber and approve of two candidates in $C \setminus \tilde{C}$.
- $\alpha = (2,0)$. In this case, we look for a matching with six edges. Note that a possible bribery cannot exist since c' has at least two points in the final election and c at most one. We distinguish three cases:

1. v_1 yields two edges in the matching. Then $\{k_1^{v_i}, k_2^{v_i}\}$ ($i = 2,3$) must be in a feasible matching (due to $b_u(c') = 1$). Even if the edge incident to b is in the matching, we have only five edges in total. Due to symmetry, we argue analogously when v_2 yields two edges in the matching.
2. v_3 corresponds to two edges in the matching. Again the other two voters may yield only the edges $\{k_1^{v_i}, k_2^{v_i}\}$ ($i = 1,2$) in a feasible matching (due to c''s upper capacity). Once again, the resulting matching has at most cardinality five.
3. Each voter v_i corresponds to one edge $\{k_1^{v_i}, k_2^{v_i}\}$ in the matching ($1 \leq i \leq 3$). Then only one edge incident to b may join the matching which has a size of four then. Again, the matching has not the required size.

Hence, we reject for guess $(2,0)$.

Example of Calculation for the Algorithm in the proof of Theorem 4.10

In this section, we consider a numerical example to illustrate the algorithm in the proof of Theorem 4.10. Suppose that we are given a set of candidates $C = \{c, c_1, c_2\}$, registered voter set $V = \{v_1, \ldots, v_{11}\}$, and unregistered voter set $W = \{w_1, \ldots, w_{10}\}$. Moreover, c is our distinguished candidate and $\ell = 3$ denotes the number of voters that may be added from W. Let the veto numbers be as follows.

$$\mathit{vetoes}_{(C,V)}(c) = 2, \quad \mathit{vetoes}_{(C,V)}(c_1) = 4, \quad \mathit{vetoes}_{(C,V)}(c_2) = 5,$$

$$\mathit{vetoes}_{(C,W)}(c) = 2, \quad \mathit{vetoes}_{(C,W)}(c_1) = 5, \quad \mathit{vetoes}_{(C,W)}(c_2) = 3.$$

In the following, let $\mathit{vetoes}_{(C,V)}(d) = |V_d|$ and $\mathit{vetoes}_{(C,W)}(d) = |W_d|$ ($d \in C$).

Our algorithm step by step computes the values a^j_{ti}, that is, the number of ways to add at most t voters vetoing the first i non-distinguished candidates such that c has fewer vetoes than the first i candidates in the final election when j vetoes from voters in W additionally count for c (recall that we do not count here the number of possibilities to add j vetoes for c from W; this will be calculated separately after running the recursion algorithm). Note that it holds $j \in \{0, \ldots, \min(\mathit{vetoes}_{(C,W)}(c), \ell) = \min(2,3) = 2\}$, $i \in \{1,2\}$, and $t \in \{0, \ldots, \ell - j\} = \{0, \ldots, 3 - j\}$. At the beginning, we initialize our procedure with $j = 0$, $t = 0$, and $i = 1$.

- $j = 0$.
  - $t = 0$.
    - $i = 1$: $a^j_{ti} = a^0_{01} = 1$ since $|V_c| + j = 2 + 0 < |V_{c_1}| = 4$ holds.
    - $i = 2$: $a^j_{ti} = a^0_{02} = 1$ since $|V_c| + j = 2 + 0 < \min(|V_{c_1}|, |V_{c_2}|) = \min(4,5) = 4$. (In other words, there is one way to a add zero vetoes for c and a total of zero vetoes for the first two non-distinguished candidates (i.e., c_1 and c_2), and c is the unique winner. Conversely, in case c_1 or c_2 would have no more vetoes than c among the voters in V, it would hold $a^0_{02} = 0$.)
  - $t = 1$.
    - $a^0_{11} = \binom{5}{1} = 5$ since $t = 1 > |V_c| + j - |V_{c_1}| = 2 + 0 - 4 = -2$.
    - $a^0_{12} = \sum_{s=\max(2+0-5+1,0)}^{\min(3,1)} \binom{3}{s} a^0_{1-s,1} = 1 \cdot a^0_{11} + 3 \cdot a^0_{01} = 1 \cdot 5 + 3 \cdot 1 = 8$. (Remind that the value a^0_{12} denotes the number of ways to (1) add zero vetoes for c from W, (2) select one voter in W vetoing c_1 or c_2 (generally, the added voters distribute their vetoes among the first i non-distinguished candidates), and (3) c has fewer vetoes than c_1 and c_2. The sum expresses that there are $\binom{3}{0} \cdot a^0_{11}$ ways to add zero vetoes for c_2 and a total of one veto for the first candidate. Moreover, we have $\binom{3}{1} \cdot a^0_{01}$ ways to select one voter in W vetoing c_2 and zero voters vetoing the first candidate, i.e., c_1.)
  - $t = 2$.
    - $a^0_{21} = \binom{5}{2} = 10$ because $t = 2 > |V_c| + j - |V_{c_1}| = 2 + 0 - 4 = -2$ holds.

* $a^0_{22} = \sum_{s=\max(2+0-5+1,0)}^{\min(3,2)} \binom{3}{s} a^0_{2-s,1} = \binom{3}{0} a^0_{21} + \binom{3}{1} a^0_{11} + \binom{3}{2} a^0_{01} = 1 \cdot 10 + 3 \cdot 5 + 3 \cdot 1 = 28$.

– $t = 3$.

* $a^0_{31} = \binom{5}{3} = 10$ since $t = 3 > |V_c| + j - |V_{c_1}| = 2 + 0 - 4 = -2$ holds.
* $a^0_{32} = \sum_{s=\max(2+0-5+1,0)}^{\min(3,3)} \binom{3}{s} a^0_{3-s,1} = \binom{3}{0} a^0_{31} + \binom{3}{1} a^0_{21} + \binom{3}{2} a^0_{11} + \binom{3}{3} a^0_{01} = 1 \cdot 10 + 3 \cdot 10 + 3 \cdot 5 + 1 = 56$.

• $j = 1$.

– $t = 0$.

* $a^1_{01} = 1$ because $|V_c| + j = 2 + 1 < |V_{c_1}| = 4$.
* $a^1_{02} = 1$ since $|V_c| + j = 2 + 1 < \min(|V_{c_1}|, |V_{c_2}|) = \min(4,5) = 4$.

– $t = 1$.

* $a^1_{11} = \binom{5}{1} = 5$ since $t = 1 > |V_c| + j - |V_{c_1}| = 2 + 1 - 4 = -1$ holds.
* $a^1_{12} = \sum_{s=\max(2+1-5+1,0)}^{\min(3,1)} \binom{3}{s} a^1_{1-s,1} = \binom{3}{0} a^1_{11} + \binom{3}{1} a^1_{01} = 1 \cdot 5 + 3 \cdot 1 = 8$.

– $t = 2$.

* $a^1_{21} = \binom{5}{2} = 10$ since $t = 2 > |V_c| + j - |V_{c_1}| = 2 + 1 - 4 = -1$ holds.
* $a^1_{22} = \sum_{s=\max(2+1-5+1,0)}^{\min(3,2)} \binom{3}{s} a^1_{2-s,1} = \binom{3}{0} a^1_{21} + \binom{3}{1} a^1_{11} + \binom{3}{2} a^1_{01} = 1 \cdot 10 + 3 \cdot 5 + 3 \cdot 1 = 28$.

• $j = 2$.

– $t = 0$.

* $a^2_{01} = 0$ since $|V_c| + j = 2 + 2 \not< |V_{c_1}| = 4$. (There is no way to add two vetoes for c and no veto for the first non-distinguished candidate such that c has fewer vetoes than c_1 in the resulting election. Hence, we obtain $a^2_{01} = 0$.)
* $a^2_{02} = 0$ because $|V_c| + j = 2 + 2 \not< \min(|V_{c_1}|, |V_{c_2}|) = \min(4,5) = 4$.

– $t = 1$.

* $a^2_{11} = \binom{5}{1} = 5$ due to $1 > 2 + 2 - 4 = 0$.
* $a^2_{12} = \sum_{s=\max(2+2-5+1,0)}^{\min(3,1)} \binom{3}{s} a^2_{1-s,1} = \binom{3}{0} a^2_{11} + \binom{3}{1} a^2_{01} = 1 \cdot 5 + 3 \cdot 0 = 5$.

Now we are able to compute $N(C,V,W,c,\ell) = \sum_{j=0}^{\min(3,2)} \binom{2}{j} \sum_{t=0}^{3-j} a^j_{t2} = \binom{2}{0}(a^0_{02} + a^0_{12} + a^0_{22} + a^0_{32}) + \binom{2}{1}(a^1_{02} + a^1_{12} + a^1_{22}) + \binom{2}{2}(a^2_{02} + a^2_{12}) = 1 \cdot (1 + 8 + 28 + 56) + 2(1 + 8 + 28) + 1(0 + 5) = 172$ as the total number of ways to add at most $\ell = 3$ voters from W such that c is the unique winner in the final election.

Calculation of the Winning Probabilities in Example 4.34

Recall that $C = \{c, c_1, c_2\}$, $V = \{v_1, \ldots, v_8\}$, $1 \leq K \leq 8$, and $\mathit{vetoes}(c) = 2$, $\mathit{vetoes}(c_1) = \mathit{vetoes}(c_2) = 3$. In this section, we derive the probabilities $\mathrm{Prob}(K)$ that a K-voter subelection yields c as the unique winner in a K-voter subelection, where $1 \leq K \leq 8$.

1. $K = 1$. We obtain a zero probability as in each subelection with one voter c_1 or c_2 is not vetoed.

2. $K = 2$. It holds $\mathrm{Prob}(2) = \frac{\binom{3}{1}\binom{3}{1}}{\binom{8}{2}} = \frac{9}{28}$ since there are $\binom{8}{2}$ possible ways to select two out of eight voters and $\binom{3}{1} \cdot \binom{3}{1} = 9$ possibilities that the lottery selects one veto for c_1 and c_2 each.

3. $K = 3$. Again, c can only be the unique winner when the lottery picks no voters vetoing c. We compute $\mathrm{Prob}(3) = \frac{\binom{3}{1}\binom{3}{2}+\binom{3}{2}\binom{3}{1}}{\binom{8}{3}} = \frac{9}{28}$. The numerator describes how often the lottery can either pick two vetoes for c_1 and one veto for c_2 or two vetoes for c_2 and one veto for c_1. In total, there are $\binom{8}{3}$ possibilities to select three out of eight voters.

4. $K = 4$. Once again, the lottery must not draw any voter vetoing c. Hence, we count the number of ways to select one veto for c_1/c_2 and three vetoes for c_2/c_1 or two vetoes for c_1 and c_2 each. This results in $\mathrm{Prob}(4) = \frac{\binom{3}{1}\binom{3}{3}+\binom{3}{2}\binom{3}{2}+\binom{3}{3}\binom{3}{1}}{\binom{8}{4}} = \frac{3}{14}$.

5. $K = 5$. In case the lottery ignores no c-veto, there is no way for c to be the unique winner. In case one veto for c is ignored by the lottery, the lottery has to leave out one veto for c_1 and c_2 each. If the lottery rules out two voters vetoing c, the third ignored voter may arbitrarily veto c_1 or c_2. As there are $\binom{8}{3}$ possibilities for the lottery to ignore three of eight voters, we obtain $\mathrm{Prob}(5) = \frac{\binom{2}{1}\binom{3}{1}\binom{3}{1}+\binom{2}{2}\binom{6}{1}}{\binom{8}{3}} = \frac{3}{7}$.

6. $K = 6$. Again, the lottery must not pick all voters vetoing c. In case the lottery ignores one voter vetoing c, it does not matter which other veto for c_1 or c_2 is left out. There is further exactly one way for the lottery to ignore two voters vetoing c. As there are $\binom{8}{6}$ ways for the lottery to ignore two of eight voters, we obtain $\mathrm{Prob}(6) = \frac{\binom{2}{1}\binom{6}{1}+\binom{2}{2}\binom{6}{0}}{\binom{8}{6}} = \frac{13}{28}$.

7. $K = 7$. There are eight different 7-voter subelections. If the voter ignored by the lottery vetoes c_1 or c_2, one of these candidates ties with c and consequently c is not the only winner. Hence, c is the winner of a 7-voter subelection if and only if the lottery omits one voter vetoing c. This leads to $\mathrm{Prob}(7) = \frac{2}{8} = \frac{1}{4}$.

8. $K = 8$. As the only 8-voter subelection is the election (C,V) itself and as c the winner in this election, we obtain $\mathrm{Prob}(8) = 1$.

Construction of K-Voter Subelections with unique-winner c in Example 4.35

In Example 4.35, our input is an election (C,V) with $m = 8$ candidates in candidate set $C = \{c, c_1, \ldots, c_7\}$, $n = 9$ voters in voter set $V = \{v_1, \ldots, v_9\}$, and distinguished candidate c. Let $\mathscr{F}$=2-Approval and the unique-winner model be the underlying winner model. The voters in V are defined as follows:

1. v_i approves of c and c_i ($1 \leq i \leq 3$).
2. $v_4, \ldots, v_6$ approve of c_4 and c_5.
3. $v_7, \ldots, v_9$ approve of c_6 and c_7.

Let K, $1 \leq K \leq 9$ be the lot size. For $K = 1$, c is not a possible winner as even if the lottery picks a voter approving of c, this voter approves of one c_i, $1 \leq i \leq 3$, as well.

For $2 \leq K \leq 3$, c is the only winner when the lottery draws the first K voters since all selected voters approve of c, but no other candidate reaches a full approval score restricted to the voters picked by the lottery.

Likewise, for $K \in \{4,5\}$, the lottery may draw the first K voters and again c the unique winner with three points: c_1, c_2, and c_3 have one point each, while c_4 and c_5 have one (for $K = 4$) or two points (for $K = 5$) each.

If $K = 6$, the subelection $(C, V \setminus \{v_6, v_8, v_9\})$ makes c the unique winner. In a similar fashion, c is the only winner of the 7-voter subelection $(C, V \setminus \{v_6, v_9\})$. In both cases, c has three approvals, c_1, c_2, and c_3 are approved by one voter each, and c_4 and c_5 have two approvals each. For $K = 6$, both c_6 and c_7 have one point, while for $K = 7$, they have two points each.

For $K = 8$, the lottery draws all three voters in $\{v_4, v_5, v_6\}$ or $\{v_7, v_8, v_9\}$, for it may ignore only one voter. In both cases, some candidate in $\{c_4, \ldots, c_7\}$ ties with c.

For $K = 9$, all votes are accounted for and c and the candidates $c_4, \ldots, c_7$ are tied for the victory with three approvals each.

As a combined result, c is a possible winner if and only if $2 \leq K \leq 7$.