On the Parameterized Complexity of Manipulating Pairwise Voting Rules

Rohit Vaish
Indian Institute of Science
India
rohit.vaish@csa.iisc.ernet.in

Neeldhara Misra
Indian Institute of Technology Gandhinagar
India
mail@neeldhara.com

ABSTRACT

Pairwise voting rules are a generalization of the standard voting rules where instead of a ranked list, each voter provides a set of pairwise comparisons between the candidates and the voting rule picks a unique winner based on these preferences. In this paper, we study the parameterized complexity of manipulation of pairwise voting rules by a single manipulator when the votes are unweighted. The manipulator faces a graph orientation problem where the vertices correspond to the candidates and the edges correspond to the pairwise comparisons that the manipulator is allowed to make. We study the effect of various structural parameters associated with this graph on the computational complexity of the manipulation problem and provide a comprehensive classification of the complexity landscape. We also introduce a new parameter called diversity which is shown to have useful algorithmic implications.

General Terms

Algorithms, Economics, Theory

Keywords

Social Choice Theory, Voting, Manipulation, Pairwise Preferences, Parameterized Complexity

1. INTRODUCTION

One of the most well-studied questions in social choice theory [1] concerns the problem of manipulation of voting rules: given the votes of all the other voters, is it possible for a strategic voter (namely the *manipulator*) to make a preferred candidate win the election by casting a possibly non-truthful vote? Unfortunately, the celebrated Gibbard-Satterthwaite theorem [2, 3] states that strategic voting is unavoidable for any voting rule that is non-dictatorial and under which each of the three or more candidates has some chance of winning.

Inspired by the work of Bartholdi, Tovey and Trick [4], a large body of follow-up work has studied when and how the computational difficulty of finding a manipulative vote can be used as an effective workaround to this impossibility (see [5] for a survey on this topic). Much of this literature focuses on voting rules that aggregate preferences provided in the form of *complete rankings* over the entire set of candidates. This assumption, however, breaks down for large-scale settings like recommender systems that involve extremely large candidate sets (e.g. movies, products, webpages etc.). In such settings, it is much more practical to elicit *partial preferences* from the users in the form of top-k preferences [6, 7, 8], partial orders [9] etc. *Pairwise preferences* are the simplest form of partial preferences where each voter is only required to provide a set of pairwise comparisons between the candidates, without either having to compare all pairs of candidates (i.e. possibly incomplete) or provide a transitive vote (i.e. possibly cyclic).

Recent work [10] has studied the problem of manipulation of *pairwise voting rules* (i.e. voting rules that aggregate pairwise preferences) from both axiomatic and computational perspectives. It has been shown that while the impossibility of designing reasonable, non-manipulable voting rules extends to the much larger domain of pairwise preferences, computational complexity can once again provide a worst-case barrier against manipulation. The goal of our study is to develop a deeper understanding of the computational complexity results in [10] using the toolkit of *parameterized complexity analysis* [11, 12, 13, 14]. This involves a fine-grained analysis of the running time in terms of the various natural parameters associated with the problem, as opposed to a coarse dependence on the size of the input as in the classical setting [15].

Specifically, we follow the framework of [10] where the manipulator is presented with an undirected graph (called the *action space*), where each vertex corresponds to a candidate and the edges correspond to the pairs of candidates that the manipulator is allowed to compare. The task of the manipulator is to orient some or all of these edges (via votes of the form $A \succ B$, $B \succ A$ or *skip*) in order to make a favorite candidate win the election. We study how some of the natural structural parameters associated with this graph (like *vertex cover*, *feedback vertex set*, *maximum degree*, *treewidth* etc.) affect the computational complexity of the manipulation problem. Our results provide a comprehensive classification of the complexity landscape for all combinations of these parameters (see Table 1). An interesting feature of our study is the introduction of a parameter called *diversity* which, in conjunction with other structural parameters, explains a complete transition in the complexity of the manipulation problem from computational tractability (i.e. FPT, XP) to intractability (i.e. W-hardness, para-NP-hardness).

Appears at: 3rd Workshop on Exploring Beyond the Worst Case in Computational Social Choice. Held as part of the 15th International Conference on Autonomous Agents and Multiagent Systems. May 9th-10th, 2016. Singapore.

	vc	pw	fvs	tw	$\emptyset$
d	FPT	W[1]-hard and XP			para-NP-complete
Δ	FPT				
$\emptyset$	para-NP-complete				

Table 1: *Parameterized complexity results for* pBorda-Manipulation *under the conditions specified by the corresponding combination of parameters. The notation $\emptyset$ is used to enable the consideration of singleton parameters. Merged cells indicate combined parameters (refer Section 2 for relevant definitions).*

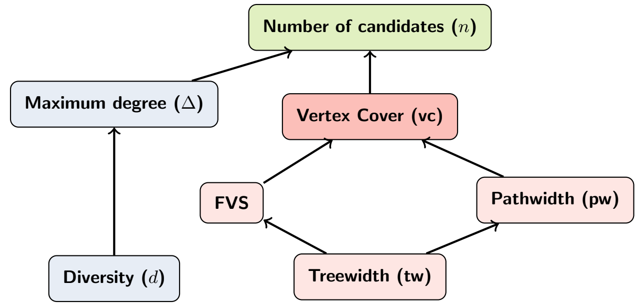


Figure 2: *A Hasse diagram depicting the relationship between the parameters. Each arrow is directed from a smaller parameter to a larger one. The implications for parameterized tractability (FPT, XP) propagate upwards along the figure in the direction of the arrows while intractability (W-hardness) propagates in the opposite direction.*

Contributions

Our contributions are as listed below. We refer the reader to Table 1 for a summary of the results and to Section 2 for relevant definitions. Figure 2 shows the relationship among the various parameters considered in this study. All of our results focus on a specific pairwise voting rule called *pairwise Borda* (pBorda) which is defined later. The computational problem corresponding to the manipulation of pBorda rule is referred to as pBorda-Manipulation. Also note that all parameters considered by us in this study are defined with respect to the *action space* $\mathcal{A}$ of the manipulator.

1. We show that pBorda-Manipulation is efficiently solvable when $\mathcal{A}$ has bounded *treewidth*, subject to bounded *maximum degree* parameter. This extends the tractability result for $\mathcal{A}$ = tree graph shown in [10].
2. We also define a new parameter called *diversity* which, in combination with the parameter *vertex cover*, provides tractability even on instances where *maximum degree* can be unbounded (i.e. grow with the number of candidates).
3. Finally, we show *computational barriers* to extending the above tractability results to *any* other combination of the parameters listed in Table 1.

Organization of this paper.

We provide the relevant definitions and notation in Section 2 and describe our results and proof techniques in Section 3. We survey the related literature in Section 4 and conclude with some directions for future work in Section 5.

2. PRELIMINARIES

Our terminology and notation closely follow that of [10]. Let $[n] = \{1, 2, \dots, n\}$ denote the set of *candidates* and $\mathcal{U} = \{u_1, u_2, \dots, u_m\}$ denote the set of *voters* in an election.

Pairwise preferences and pairwise voting rules.

Let $\succ_u \subseteq [n] \times [n]$ denote the binary relation indicating the preferences of voter u, so that $i \succ_u j$ indicates that voter u prefers candidate i over candidate j. For each pair of candidates i, j and each voter u, we can have exactly one of $i \succ_u j$, $j \succ_u i$ or neither (i.e. voter u *skips* the comparison between i and j). We let $\mathcal{R}$ denote the set of all such anti-symmetric and irreflexive binary relations on $[n]$; and let $\Pi = (\succ_{u_1}, \succ_{u_2}, \cdots, \succ_{u_m}) \in \mathcal{R}^m$ denote the *pairwise preference profile* of the voters.

A *pairwise voting rule* r maps a pairwise preference profile $\Pi \in \cup_{k=1}^{\infty} \mathcal{R}^k$ to a unique candidate $r(\Pi) \in [n]$. Given a preference profile $\Pi \in \mathcal{R}^m$ and a pair of candidates i, j, let $m_{ij}(\Pi)$ denote the number of voters who strictly prefer candidate i over candidate j, i.e. $m_{ij}(\Pi) = \sum_{k=1}^{m} \mathbf{1}(i \succ_{u_k} j)$ where $\mathbf{1}(.)$ is the indicator function. A *score-based pairwise voting rule* is any pairwise voting rule r for which there exists a (natural) scoring function $\mathbf{s} : \cup_{k=1}^{\infty} \mathcal{R}^k \to \mathbb{R}^n$ such that $r(\Pi)$ is the highest-scoring candidate according to $\mathbf{s}(\Pi)$ under some fixed tie-breaking rule. That is, $r(\Pi) = T(\arg\max_i \, s_i(\Pi))$ for some tie-breaking rule $T : 2^{[n]} \setminus \{\emptyset\} \to [n]$ satisfying $T(S) \in S$ for all non-empty $S \subseteq [n]$. Some examples of score-based pairwise voting rules are as follows:

(i) *Pairwise Borda Rule* (pBorda) [16]: The pBorda score of candidate i under preference profile Π is given by[1]:

$$s_i^{\text{pBorda}}(\Pi) = \sum_{j=1}^{n} \frac{m_{ij}(\Pi)}{m_{ij}(\Pi) + m_{ji}(\Pi)}.$$

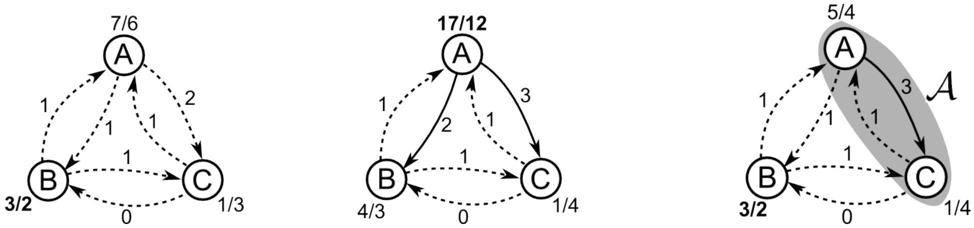
(ii) *Copeland$^\alpha$ Rule* [17]: The Copeland$^\alpha$score ($\alpha \in [0, 1]$) of candidate i under preference profile Π is given by:

$$s_i^{\text{Copeland}^\alpha}(\Pi) = \sum_{j=1}^{n} \mathbf{1}\big(m_{ij}(\Pi) > m_{ji}(\Pi)\big) + \alpha \cdot \mathbf{1}\big(m_{ij}(\Pi) = m_{ji}(\Pi)\big).$$

Manipulation of pairwise voting rules.

A pairwise voting rule r is said to be *manipulable* if there exists a pair of profiles $\Pi = (\succ_{u_1}, \dots, \succ_{u_m}), \Pi' = (\succ_{u_1}, \dots, \succ_{u_{m-1}}, \succ'_{u_m}) \in \mathcal{R}^m$ differing only in the preference of voter u_m such that $r(\Pi') \succ_{u_m} r(\Pi)$. That is, voter u_m (called *the manipulator*) strictly prefers the new outcome over the old one. The corresponding computational problem, referred to as r-Manipulation, is defined as follows:

[1]where we adopt the convention $0/0 = 0$.



(a) *Before manipulation* **(b)** *After manipulation* **(c)** *After restricted manipulation*

Figure 1: *An illustration of the election instance in Example 2.1. (a) Each vertex of the multigraph represents a candidate and each dashed edge represents the number of voters with that preference (e.g. two voters prefer $A \succ C$). (b) The pairwise comparisons made by the manipulator are represented by solid edges and the* pBorda *score of the winning candidate is indicated in boldface. (c) The restricted action space of the manipulator ($\mathcal{A} = \{(A, C)\}$) is shaded in grey.*

Definition 2.1. r-Manipulation

> ***Instance:*** *A tuple $\langle \Pi, i^*, \mathcal{A}, \texttt{pref-type} \rangle$ where $\Pi \in \mathcal{R}^{m-1}$ is the preference profile of the non-manipulators $(u_1, u_2, \ldots, u_{m-1})$, $i^* \in [n]$ is the distinguished candidate, $\mathcal{A} \subseteq \binom{[n]}{2}$ is the set of pairwise comparisons that the manipulator is allowed to make and* `pref-type` $\in$ {strict+acyclic, strict, acyclic, unrestricted} *is the preference constraint with respect to $\mathcal{A}$.*
>
> ***Question:*** *Does there exist a vote $\succ_{u_m}$ over $\mathcal{A}$ satisfying* `pref-type` *such that $r((\Pi, \succ_{u_m})) = i^*$?*

Here $\mathcal{A} \subseteq \binom{[n]}{2}$ denotes the *action space* of the manipulator i.e. the pairs of candidates that the manipulator is allowed to vote over. Alternately, no pair of candidates outside $\mathcal{A}$ can be compared by the manipulator. The parameter `pref-type` indicates whether the preferences of the manipulator over $\mathcal{A}$ are required to be *strict* (skipping comparisons is not allowed), *acyclic* (directed cycles of the form $1 \succ_u 2, 2 \succ_u 3, 3 \succ_u 1$ etc. are not allowed), *strict+acylic* (both *strict* and *acylic*) or *unrestricted* (no such restriction). The computational complexity of r-Manipulation was studied for various settings of the inputs $\mathcal{A}$ and `pref-type` in [10]. In this paper, however, we only focus on problems where `pref-type` = unrestricted and leave the study for other settings of `pref-type` as a direction for future work. The following example from [10] illustrates the role of the space $\mathcal{A}$ in the manipulation problem.

Example 2.1 (The role of action space $\mathcal{A}$). *Consider the election setting shown in Figure 1a, where the* pBorda *scores of the candidates A, B & C respectively are 7/6, 3/2 & 1/3 and B is the* pBorda *winner. Suppose we now add the manipulator u_4 to this election whose favorite candidate is A. Observe that if the manipulator casts the vote $\{(A \succ B), (A \succ C)\}$ (see Figure 1b), the new* pBorda *scores for A, B & C will be 17/12, 4/3 & 1/4 respectively and A becomes the winner. Thus, the answer to* pBorda-Manipulation *for this election instance is YES when $\mathcal{A} = \{(A, B), (A, C)\}$ or $\mathcal{A} = \{(A, B), (A, C), (B, C)\}$. If, however, the manipulator is allowed to compare only the candidates A and C (that is, $\mathcal{A} = \{(A, C)\}$), then despite voting in favor of A, the manipulator cannot make A win (Figure 1c). Therefore, the answer to* pBorda-Manipulation *is NO when $\mathcal{A} = \{(A, C)\}$.*

Excess scores.

The excess score of a candidate i is the amount by which the score of i exceeds the score of the distinguished candidate i^* in a given election. For instance, in Figure 1c, the excess pBorda scores of candidates B and C (with respect to distinguished candidate A) are $1/4$ and -1 respectively. Hence, r-Manipulation for a score-based voting rule r can be restated as finding a vote for the manipulator such that the final excess scores of all candidates are zero or less.

Vote configuration.

We will often use a shorthand of the form 1:3 for a pair of candidates (i, j) to denote that one voter votes $i \succ j$ while three other voters vote $j \succ i$. We will refer to 1:3 (or more generally $a : b$ for non-negative integers a, b) as the *vote configuration* between i and j.

Parameterized Complexity.

A *parameterized problem* is denoted by a pair $(Q, k) \subseteq \Sigma^* \times \mathbb{N}$. The first component Q is a classical language and the second component k is a number (called the *parameter*). Such a problem is called *fixed–parameter tractable* (FPT) if there exists an algorithm that decides it in time $\mathcal{O}(f(k) n^{\mathcal{O}(1)})$ on instances of size n.

Just as NP-hardness is used as evidence that a problem probably is not polynomial time solvable, there exists a hierarchy of complexity classes above FPT, and showing that a parameterized problem is hard for one of these classes is considered evidence that the problem is unlikely to be fixed-parameter tractable. The main classes in this hierarchy are

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

where a parameterized problem belongs to the class XP if there exists an algorithm for it with running time bounded by $n^{g(k)}$ for some computable function g. We refer the reader to [11, 12, 13, 14] for further details.

A parameterized problem is said to be *para-NP-complete* if it is NP-complete even for constant values of the parameter. A classic example of a para-NP-complete problem is Graph Coloring parameterized by the number of colors [18] — recall that it is NP-complete to determine if a graph can be properly colored with three colors. Observe that a para-NP-complete problem does not belong to XP unless $P = NP$.

For any pair of parameterized problems A and B, we say that A is (uniformly many:1) *FPT-reducible* to B if there exist functions $f, g : \mathbb{N} \to \mathbb{N}$, a constant $\alpha \in \mathbb{N}$ and an algorithm Φ which transforms an instance (x, k) of A into an instance $(x', g(k))$ of B in time $f(k) \cdot |x|^{\alpha}$ so that $(x, k) \in A$ if and only if $(x', g(k)) \in B$. A convenient way of showing that a problem is W[1]-hard is via an FPT reduction from a known W[1]-hard problem. Hence, in the above definition, if the problem A is known to be W[1]-hard in parameter k and there exists an FPT reduction from A to B, then B is W[1]-hard in the parameter $g(k)$.

Parameters used in this study.

Let $G = (V, E)$ denote a simple and undirected graph.

Maximum degree (Δ): The *maximum degree* of G is the maximum number of edges incident on any vertex of G.

Vertex Cover (**vc**): A set of vertices $V' \subseteq V$ is a *vertex cover* of G if for every edge $(u, v) \in E$, either $u \in V'$ or $v \in V'$ or both.

Feedback Vertex Set (**fvs**): A *feedback vertex set* of a graph is a set of vertices whose removal makes the graph acyclic.

Tree decomposition: A tree decomposition of a graph G is a tuple $\mathcal{T} = (T, \{B_t\}_{t \in V(T)})$ where T is a tree and each node t of T is assigned a set of vertices $B_t \subseteq V$ (called a *bag*) such that the following hold: (i) for each vertex $v \in V$, there exists a node t such that $v \in B_t$ (alternately, $\cup_{t \in V(T)} B_t = V$); (ii) for each edge $(u, v) \in E$, there exists a node t such that $u \in B_t$ and $v \in B_t$, and (iii) for each vertex $v \in V$, the set of nodes $\{t \in V(T) : v \in B_t\}$ forms a connected subtree of T. Here $V(T)$ is the vertex set of the tree T. The *width* of a tree decomposition $\mathcal{T} = (T, \{B_t\}_{t \in V(T)})$ equals $\max_{t \in V(T)} |B_t| - 1$, i.e. size of the largest bag minus one. The *treewidth* (**tw**) of a graph G is the minimum possible width of a tree decomposition of G. The notions of *pathwidth* (**pw**) and *path decomposition* are defined analogously in terms of paths.

Diversity (d): Given a preference profile Π of non-manipulators' votes, the *diversity* of the action space $\mathcal{A}$ is the maximum number of distinct score transfers that a candidate can witness due to a single pairwise comparison made by the manipulator. As an example, consider the election instance shown in Figure 1a and consider the candidate A in particular. Assuming that $\mathcal{A}$ = complete graph, the manipulator can make a pairwise comparison between any of the three pairs (A, B), (B, C) or (C, A). If the manipulator compares the pair (A, B), then the pBorda-score of candidate A can change by $+1/6$, 0 or $-1/6$ respectively, depending on whether the manipulator votes $A \succ B$, 'skip' or $B \succ A$. This can be concisely represented as a *score-transfer vector* $(+1/6, 0, -1/6)$. Similarly, the score transfer vector for candidate A for a comparison involving (A, C) or (B, C) is $(+1/12, 0, -1/6)$ or $(0, 0, 0)$ respectively. Since there are three different *kinds* of such vectors, the *diversity* for candidate A is three. The *diversity of an instance* is the maximum diversity witnessed by any candidate. Notice that for a given election, *diversity* can be $\Theta(n)$ under the pBorda rule while the same for Copeland$^{\alpha}$ is $\mathcal{O}(1)$ due to the limited types of score exchanges permitted under the definition of Copeland voting rule. For any pairwise voting rule where a pairwise comparison by the manipulator can only affect the scores of the two candidates involved (examples include pBorda and Copeland$^{\alpha}$), *diversity* is at most the *maximum degree* Δ.

Elimination problem in sports.

The sports elimination problem [19] asks whether a team i^* can still win a sports competition, given the current scores of the teams and the set of games to be played between them. Sports competitions are often scored according to a *scoring system*, which specifies how many points are awarded to the home and the away teams depending on the outcome of a game between them. For example, the well-known European football scoring system, denoted by $\mathrm{S} = [(3, 0), (1, 1), (0, 3)]$, awards 3 points for win, 1 point for draw and 0 for loss, regardless of the home-away distinction. Similarly, the system $\mathrm{S} = [(3, 0), (1, 2), (0, 3)]$ provides an extra point to an away team under a draw outcome. The computational problem corresponding to the above question, called S-ELIMINATION, is defined as follows [20]:

DEFINITION 2.2. S-ELIMINATION

Instance: *A tuple $\langle \mathbf{s}, i^*, \mathcal{G} \rangle$ where $\mathbf{s} = (s_1, s_2, \ldots, s_N)^T$ is the vector of current scores of the N teams, $i^* \in [N]$ is a distinguished team and $\mathcal{G} \subseteq \binom{[N]}{2}$ is the set of remaining games between the teams.*

Question: *Does there exist an assignment of outcomes for the games in $\mathcal{G}$ such that i^* ends up with the (joint) highest total score among all teams under the scoring system* S*?*

Partition.

DEFINITION 2.3. PARTITION

Instance: *A multiset $A = \{a_1, a_2, \ldots, a_N\}$ of N positive integers.*

Question: *Does there exist a partition of A into the sets A_1 & A_2 such that $\sum_{a_i \in A_1} a_i = \sum_{a_j \in A_2} a_j = \frac{1}{2} \sum_{a_k \in A} a_k$?*

PARTITION is a well-known NP-complete problem [18]. We assume without loss of generality that $a_1 \leq a_2 \leq \cdots \leq a_N$.

Capacitated Dominating Set.

DEFINITION 2.4. CAPACITATED DOMINATING SET

Instance: *A triple $\langle G, c, k \rangle$ where $G = (V, E)$ is a graph, $c : V \to \mathbb{N}$ is a capacity function for the vertices of G and k is a positive integer.*

Question: *Does there exist a set of vertices $V' \subseteq V$ of size at most k in G such that each vertex $v \in V \setminus V'$ is adjacent to some vertex $v' \in V'$ and no vertex $v' \in V'$ is adjacent to more than $c(v')$ vertices in $V' \setminus V$?*

CAPACITATED DOMINATING SET was shown to be W[1]-hard when simultaneously parameterized by the *treewidth* and solution size k [21]. In fact, the problem remains W[1]-hard when simultaneously parameterized by the *pathwidth* and the size of the *feedback vertex set* of the graph G, even on instances with only constantly many distinct capacities[2].

[2] This can be shown by carrying out the reduction in [21] while starting from MULTICOLORED CLIQUE on regular graphs and observing that the parameters *pathwidth*, *treewidth* and *feedback vertex set* of the reduced CAPACITATED DOMINATING SET instance are all $\mathcal{O}(k^4)$ in size.

3. OUR RESULTS AND TECHNIQUES

Our classification result for the parameterized complexity of pBorda-Manipulation for any combination of the considered parameters is summarized by Theorem 3.1 and Table 1. We assume throughout that `pref-type`=unrestricted.

In the parameterized studies of computational problems that arise in the context of voting, a commonly used parameter is the *number of candidates* (n) [22, 23, 24, 25, 26, 27]. We observe that for any pairwise voting rule that is easy to evaluate, the problem of manipulation by a single manipulator is trivially FPT for this choice of parameter, because even a brute-force search over all possible votes of the manipulator will yield the desired running time (i.e. $\mathcal{O}(3^{n^2})$). The other natural choice of parameter is the *number of voters*. However, we know from [10] that pBorda-Manipulation is NP-complete *even with twelve non-manipulators*. Given the extreme behaviors on the two obvious choices of parameters, we turn to the action space of the manipulator $\mathcal{A}$ and try to understand how the problem complexity is influenced by parameters associated with the structure of $\mathcal{A}$.

We start by recalling the result in [10] which states that that pBorda-Manipulation is efficiently solvable when $\mathcal{A}$ is a tree/forest/graph with maximum degree two. Given this result, we follow the "distance from triviality" approach in parameterized analysis [28] and consider parameters that measure how *far* $\mathcal{A}$ is from the class of tractable instances i.e. degree of closeness to being a tree or a forest. This motivates the study of parameters like *treewidth* (**tw**), *feedback vertex set* (**fvs**) and *maximum degree* (Δ); and upper/lower bounds on these parameters like *pathwidth* (**pw**) and *vertex cover* (**vc**) (refer Section 2 for formal definitions). Interestingly, a similar set of parameters was recently used in the parameterized complexity analysis of the closely-related S-Elimination problem [29], and studying their influence on the complexity of pBorda-Manipulation allows us to compare the complexity landscapes of the two problems, as we will see.

Our first set of results shows that the manipulation problem is, somewhat surprisingly, *para-NP-complete* for (i) the *maximum degree* parameter, and (ii) *any combination of the parameters in* {**vc**, **fvs**, **pw**, **tw**}. This already establishes a contrast with S-Elimination which was shown to be in XP when parameterized by the *treewidth* of the graph formed by the set of remaining games [29].

On the other hand, pBorda-Manipulation is FPT when simultaneously parameterized by *maximum degree* and *any combination of the other parameters*. We ask if there is a *natural* parameter that is, in general, smaller than *maximum degree*, but that can still provide tractability when combined with some of the other structural parameters. We discover an answer in the form of a novel parameter called *diversity* (d), which is a measure of how many different types of score exchanges the manipulator encounters for any candidate. Unfortunately, it turns out that pBorda-Manipulation is NP-complete on graphs with constant *diversity*; in fact, it remains NP-complete even when the sum of *diversity* and *maximum degree* is bounded by a constant [10]. On the positive side, we show that *diversity*, when combined with *vertex cover*, leads us to an FPT algorithm, while we obtain XP algorithms by combining it with any of the other parameters in {**fvs**, **pw**, **tw**}. We do not expect to improve this XP result, as the problem remains W[1]-hard in those cases.

We now state our main result (Theorem 3.1) that summarizes the findings described above (see also Table 1).

Theorem 3.1. *Let $\mathcal{P} = \{$**vc**, **pw**, **fvs**, **tw**, $\Delta, d\}$ denote the set of parameters defined over the action space $\mathcal{A}$ of an instance of* pBorda-Manipulation. *Let $\mathcal{X}$ denote the set $\{$**vc**, **pw**, **fvs**, **tw**$\}$ and $\mathcal{Y}$ denote the set $\{d, \Delta\}$. Then*

1. *For any $\mathcal{Q} \subseteq \mathcal{X}$ or $\mathcal{Q} \subseteq \mathcal{Y}$,* pBorda-Manipulation *is* NP-complete *even when the sum of all parameters in $\mathcal{Q}$ is bounded by a constant.*
2. *For all $\mathcal{Q} \subseteq \mathcal{P}$,* pBorda-Manipulation *parameterized by $\mathcal{Q}$ is in XP if $\mathcal{Q}$ contains d along with any element of $\mathcal{X}$. Further, the problem is* FPT *if $\mathcal{Q}$ contains Δ along with any element of $\mathcal{X}$, or if it contains both d and* **vc**.
3. *In the remaining case when $\mathcal{Q} \subseteq \mathcal{P}$ does not contain either Δ or* **vc**, pBorda-Manipulation *is* W[1]-hard *parameterized by $\mathcal{Q}$, even on instances where d is bounded by a constant.*

We briefly summarize these results and their implications.

(i) We show that pBorda-Manipulation remains NP-complete even for instances where $\mathcal{A}$ has a *vertex cover* of size two (Theorem 3.2). Since a bound on the size of the *vertex cover* implies a bound on the size of the *feedback vertex set*, *pathwidth* and *treewidth*, we have NP-completeness of pBorda-Manipulation even when parameterized by all parameters in $\mathcal{X}$ combined. Together with the result from [10] showing NP-hardness of pBorda-Manipulation on instances of *maximum degree* $\Delta = 3$ (and therefore *diversity* $d \leq 3$), this implies statement 1 of Theorem 3.1.

(ii) We use dynamic programming over tree decompositions to show that pBorda-Manipulation is FPT when parameterized by *maximum degree* and *treewidth* (Theorem 3.4). Since all other parameters in $\mathcal{X}$ are larger than *treewidth*, this gives an FPT result when Δ is combined with any subset of parameters in $\mathcal{X}$. Similarly, we use Lenstra's result [30] on Integer Linear Programming being FPT in the number of variables to show that pBorda-Manipulation is FPT when simultaneously parameterized by the *vertex cover* and *diversity* of $\mathcal{A}$ (Theorem 3.5). These two results together imply statement 2 of Theorem 3.1.

(iii) Finally, we show that pBorda-Manipulation is W[1]-hard when simultaneously parameterized by the *feedback vertex set* and *pathwidth* of $\mathcal{A}$ via an FPT-reduction from Capacitated Dominating Set [21]. This proves statement 3 part of Theorem 3.1.

We now provide formal statements and proofs for the results stated above. Our first result shows that pBorda-Manipulation is para-NP-complete in *vertex cover*.

Theorem 3.2. pBorda-Manipulation *is* NP-complete *when $\mathcal{A}$ is a general graph with a vertex cover of size two and* `pref-type` = *unrestricted.*

Proof. The problem is clearly in NP. We show NP-hardness by reduction from Partition.

Construction of the reduced instance: Given an instance $A = \{a_1, a_2, \ldots, a_N\}$ of Partition, we construct an instance $\langle \Pi, i^*, \mathcal{A}, \texttt{pref-type}\rangle$ of pBorda-MANIPULATION as follows: the *set of candidates* consists of (i) the *selector* candidates X and Y, (ii) a candidate i for each positive integer $a_i \in A$ (called the *integer* candidates), (iii) the *distinguished candidate* i^* and (iv) the *dummy candidates* $D_1, D_2, \ldots, D_{4N}$ (hence $n = 5N+3$ candidates overall). The *action space* $\mathcal{A}$ is the complete bipartite graph between the *selectors* and the *integer* candidates. That is, $\mathcal{A} = \{\cup_{i\in[N]}\{(X,i)\cup(Y,i)\}\}$. The *set of voters* consists of $2Q$ non-manipulators (where $Q = (2a_N+1)\cdot(2a_N+2)$) and one manipulator. The *votes of the non-manipulators* between the *selectors* and the *integer* candidates are set up in order to ensure that the score transfers resulting from the manipulator's vote are a_i/Q (if the manipulator votes $X\succ i$ or $Y\succ i$) and $(2a_N+1-a_i)/Q$ (if the manipulator votes $i\succ X$ or $i\succ Y$). Specifically, for each $i \in [N]$, both the candidate pairs (i,X) and (i,Y) are in $a_i : (2a_N+1-a_i)$ configuration. The *votes involving dummy candidates* are set up as follows: for each $i \in [N]$, the pair (i^*, D_i) is in $(2a_N+1-a_i) : a_i$ configuration while the pair (i^*, D_{N+i}) is in $a_i : (2Q-a_i)$ configuration. For each $i \in [N]$, the pair (i, D_{2N+i}) is in $(2a_N+1-3a_i) : 3a_i$ configuration while the pair (i, D_{3N+i}) is in $2a_i : (2Q-2a_i)$ configuration. Finally, for each $i \in [N]$ and each $k \in [N]\setminus i$, the pair (i, D_{2N+k}) is in $(2a_N+1-a_k) : a_k$ configuration while the pair (i, D_{3N+k}) is in $a_k : (2Q-a_k)$ configuration.

It is easy to check that the excess score of each *integer* candidate i after this construction is $a_i/2Q$, while that of each *selector* is $\frac{-1}{2Q}\sum_{a_k\in A} a_k$. Also note that the *selector* vertices constitute a *vertex cover* of $\mathcal{A}$ of size two.

Equivalence of solutions: ($\Rightarrow$) Suppose there exists a partition of A into the sets A_1 and A_2. A valid manipulative vote can be constructed from this partition as follows: for each $i \in [N]$, the manipulator votes $X\succ i$ if $a_i \in A_1$ or $Y\succ i$ if $a_i \in A_2$ and skips all other comparisons. The final excess score of each *integer* candidate i is negative, since $\frac{a_i}{2Q} - \frac{a_i}{Q} < 0$. The final excess score for each *selector* equals 0 due to the partition property, making i^* the winner.

($\Leftarrow$) Suppose there exists a valid manipulative vote that makes i^* win. Then, without loss of generality, each *integer* candidate i must lose at least one of its two pairwise comparisons in $\mathcal{A}$ in order to get rid of its positive excess score (Observation 1). Similarly, no *integer* candidate i can win either of its pairwise comparisons against any of the *selectors* or otherwise it accumulates an excess that it cannot offload any further (Observation 2). Observation 1 implies that the combined pBorda score that gets transferred from the *integer* candidates to the *selectors* is at least $\sum_{a_k\in A}\frac{a_k}{Q}$. Observation 2 implies that no pBorda score gets transferred in the reverse direction. Since the *selectors* can together handle an influx of at most $\sum_{a_k\in A}\frac{a_k}{Q}$, each *integer* candidate i must lose to exactly one of the *selectors* while the other comparison is skipped. A partition can now be naturally inferred from such a vote. □

REMARK 3.1. *An implication of Theorem 3.2 is a separation of the problems of* pBorda-MANIPULATION *and* S-ELIMINATION *in terms of their computational complexity. As mentioned earlier,* S-ELIMINATION *was shown to be in XP when parameterized by the treewidth of the graph formed by the set of remaining games [29] while* pBorda-MANIPULATION *is para-NP-complete in the same parameter. Hence,* pBorda-MANIPULATION *is necessarily harder than* S-ELIMINATION *unless $P = NP$.*

Our next result establishes the W[1]-hardness of pBorda-MANIPULATION in terms of the size of the *feedback vertex set* and the *pathwidth* of $\mathcal{A}$, even on instances where the *diversity* of $\mathcal{A}$ is bounded by a constant.

THEOREM 3.3. pBorda-MANIPULATION *is* W[1]-hard *when simultaneously parameterized by feedback vertex set and pathwidth of $\mathcal{A}$ when $\mathcal{A}$ = general graph with constant diversity and* $\texttt{pref-type}$ = *unrestricted.*

PROOF. We show an FPT reduction from CAPACITATED DOMINATING SET. Recall from Section 2 that CAPACITATED DOMINATING SET is W[1]-hard when simultaneously parameterized by the *feedback vertex set* and *pathwidth* of the input graph even on instances with only a constant number of distinct capacities.

Construction of the reduced instance: Given an instance $\langle G = (V,E), c, k\rangle$ of CAPACITATED DOMINATING SET, we construct an instance $\langle \Pi, i^*, \mathcal{A}, \texttt{pref-type}\rangle$ of pBorda-MANIPULATION as follows: the *set of candidates* consists of (i) the *source* X and the *sink* Y, (ii) a candidate v_i for each vertex in G (the *vertex* candidates), (iii) a candidate e_i for each edge in G (the *edge* candidates), (iv) the *distinguished* candidate i^* and (v) the *dummy* candidates $D_1, D_2, \ldots, D_\ell$ where $\ell = 7|V| + 2|E| + \Delta + 2 - 2k$ and Δ is the maximum degree of graph G. Hence, $n = 8|V| + 3|E| + \Delta - 2k + 5$. The *action space* $\mathcal{A}$ is the union of all unordered pairs of candidates connected by dashed edges in Figure 3. That is,

$$\mathcal{A} = \big\{\{\cup_{i\in[|V|]}(X, v_i)\} \bigcup \{\cup_{j\in[|E|]}(e_j, Y)\} \bigcup \{\cup_{i\in[|V|], j\in[|E|]}(v_i, e_j) \text{ where } v_i \text{ is adjacent to } e_j \text{ in } G\}\big\}.$$

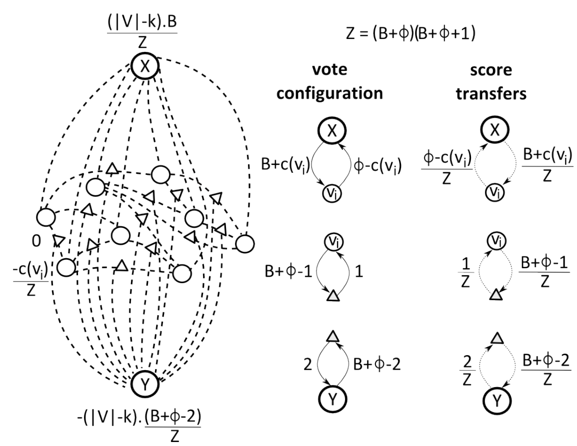


Figure 3: *This figure shows the reduced* pBorda-MANIPULATION *instance (excluding the dummy candidates and i^*) constructed from the given* CAPACITATED DOMINATING SET *instance. The action space of the manipulator $\mathcal{A}$ is shown on the left via dashed lines along with the excess* pBorda *scores. The* vertex *candidates are indicated by circles in the middle layer and the* edge *candidates are shown as triangles. The right side shows the configuration of votes of the non-manipulators and the resulting scores transfers.*

The *set of voters* consists of Z non-manipulators (where $Z = (B+\phi)\cdot(B+\phi+1)$, $B = |V| + |E| +$

$\sum_{v_i \in V(G)} c(v_i)$ and $\phi = \max_{i \in |V|} c(v_i)$) and one manipulator. The *votes corresponding to action space* $\mathcal{A}$ are set up as follows: for each $i \in [|V|]$, the candidate pair (X, v_i) is in $(B + c(v_i)) : (\phi - c(v_i))$ configuration. For each $i \in [|V|]$ and each $j \in [|E|]$ such that v_i is adjacent to e_j in G, the candidate pair (v_i, e_j) is in $(B + \phi - 1) : 1$ configuration. Finally, for each $j \in [|E|]$, the candidate pair (e_j, Y) is in $2 : (B + \phi - 2)$ configuration. We omit the description of the votes involving dummy candidates due to space limitations and remark that they are only used to calibrate the excess scores of candidates as shown in Figure 3. This finishes the construction of the election instance.

Note that the reduction is *efficient* since it uses $\mathcal{O}((|V| + |E|)^2)$ voters and $\mathcal{O}(|V| + |E|)$ candidates. Also note that the reduction is *parameter preserving*, in that if the *pathwidth* and size of the optimal *feedback vertex set* of G are w and t, then the same parameters for the action space $\mathcal{A}$ are $\mathcal{O}(w^2)$ and $(t + 2)$ respectively. Furthermore, the *diversity* of the reduced election instance is a constant since the original Capacitated Dominating Set instance only has $\mathcal{O}(1)$ distinct capacity values.

Equivalence of solutions: ($\Rightarrow$) Suppose $S \subseteq V(G)$ is a valid capacitated dominating set. Then a valid manipulative vote can be constructed as follows: first, the manipulator triggers score transfers from *source* X to *vertex* candidates in $V(G) \setminus S$ with a vote $v_i \succ X$ for all $v_i \in V(G) \setminus S$. This brings the excess score of X below zero and results in an excess of B/Z for each $v_i \in V(G) \setminus S$. Next, for each $v_i \in V(G) \setminus S$, the manipulator votes $e_j \succ v_i$ for exactly one *edge* candidate e_j that connects v_i to a candidate $v'_i \in S$ that v_i is *assigned to*. This results in a negative excess score for all *vertex* candidates in $V(G) \setminus S$ while each *edge* candidate e_j chosen above by the manipulator now acquires an excess score of $(B + \phi - 1)/Z$. Finally, for each such *edge* candidate e_j, the manipulator votes $Y \succ e_j$ and $v'_i \succ e_j$. It is easy to check that after this step no candidate in $\mathcal{A}$ has positive excess score, making i^* the pBorda winner.

($\Leftarrow$) Suppose there exists a valid manipulative vote that makes i^* win. Then, without loss of generality, X must lose against at least $(|V| - k)$ *vertex* candidates in order to offload its excess. Call this set S'. Hence, $|S'| \geq |V| - k$ and each candidate in S' acquires an excess of B/Z as a result. Next, observe that any candidate in S' can only offload its excess score to the *edge* candidates adjacent to it in G. As a result, each *edge* candidate affected in this manner (there must be at least $(|V| - k)$ such edge candidates overall) acquires an excess of $(B + \phi - 1)/Z$, that it must offload to the *sink* Y and the other *vertex* candidate adjacent to it. Hence, the *sink* Y suffers a total inflow of at least $(|V| - k) \cdot (B + \phi - 2)/Z$ from the affected *edge* candidates. By design, this is also the maximum inflow that the *sink* can handle without gaining positive excess. Therefore, the set S' must consist of exactly $(|V| - k)$ *vertex* candidates such that each candidate in S' is connected to a candidate in $V(G) \setminus S'$ via an *edge* candidate. Besides, no candidate in $V(G) \setminus S'$ can be adjacent to more than $c(v_i)$ candidates in S', or else there will be no means for this candidate to offload its own excess score. Therefore, the set $V(G) \setminus S'$ constitutes a capacitated dominating set of G. ☐

Remark 3.2. *Except for scoring systems of the form* $\mathrm{S} = \{(i, t - i) : 0 \leq i \leq t\}$ *for some* $t \in \mathbb{N}$, S-Elimination *was shown to be* W[1]*-hard in the parameters feedback vertex set and pathwidth via separate (although similar) proofs in [29]. By instantiating* S-Elimination *for* $\mathrm{S} = [(3, 0), (1, 2), (0, 3)]$ *and using the observation in [10] that* S-Elimination *becomes a special case of* pBorda-Manipulation *under such instantiation, one can alternately derive* W[1]*-hardness of* pBorda-Manipulation *in terms of the two parameters individually. By contrast, Theorem 3.3 provides a single proof for showing* W[1]*-hardness in the two parameters simultaneously.*

Our first algorithmic result shows that pBorda-Manipulation is FPT when simultaneously parameterized by the *treewidth* and *maximum degree* of $\mathcal{A}$. We show this by the standard dynamic programming procedure over a given tree decomposition [14] and omit the detailed proof due to space limitations.

Theorem 3.4. pBorda-Manipulation *is solvable in time* $\mathcal{O}\big(\Delta^{\mathcal{O}(dw^2)}(n \log m)^{\mathcal{O}(1)}\big)$*, where* Δ*,* w *and* d *denote the maximum degree, treewidth and diversity of* $\mathcal{A}$ *respectively.*

Note that since $d \leq \Delta$ for pBorda rule, the running time above is FPT in *maximum degree* Δ and the *treewidth* of $\mathcal{A}$. Further, since $\Delta \leq n$, the running time is also XP with respect to the *diversity* d and *treewidth* w of $\mathcal{A}$. We restate these observations as the following corollary.

Corollary 3.1. pBorda-Manipulation *is* FPT *when parameterized by the maximum degree and the treewidth of* $\mathcal{A}$*; and in* XP *when parameterized by the diversity and the treewidth of* $\mathcal{A}$*.*

Our next algorithmic result pertains to graphs of bounded *vertex cover* number and bounded *diversity*.

Theorem 3.5. pBorda-Manipulation *is solvable in time* $\mathcal{O}\big(f(k, d)(n \log m)^{\mathcal{O}(1)}\big)$*, where* k *and* d *denote the size of a vertex cover and diversity of* $\mathcal{A}$ *respectively and* f *is a computable function.*

Proof. The proof proceeds by partitioning the vertices of the independent set of $\mathcal{A}$ into equivalence classes based on their interactions with the vertex cover, and exploiting a size bound on the number of such equivalence classes in the subsequent ILP formulation.

Specifically, let $S \subseteq V(\mathcal{A})$ be a vertex cover of $\mathcal{A}$ of size k and let $I = V(\mathcal{A}) \setminus \mathcal{S}$ be the corresponding independent set. For any $T \subseteq S$, let $I_T \subseteq I$ denote the set of all vertices in I whose neighborhood within $\mathcal{A}$ is exactly the set T. Next, given $T = \{v_1, \ldots, v_t\} \subseteq S$ and a vector $E_T = \langle(\alpha_1, \beta_1), \ldots, (\alpha_t, \beta_t)\rangle$ consisting of pairs of non-negative integers (α_i, β_i), define the *equivalence class* $I_{T,E_T} \subseteq I_T$ such that for any pair of vertices $v_i \in T$ and $u \in I_{T,E_T}$, the candidate pair (v_i, u) is in $\alpha_i : \beta_i$ configuration with respect to the votes of non-manipulators. Note that since the diversity d of the instance is bounded, any vertex of the independent set must belong to exactly one of at most $2^k \cdot d^k$ equivalence classes.

Call a vertex $u \in I_{T,E_T}$ *safe* with respect to a vector $z \in \{-1, 0, 1\}^{|T|}$ if the excess score of u is zero or less for the following vote of the manipulator: for each $v_i \in T$, $u \succ v_i$ if $z(i) = +1$; 'skip' the comparison (u, v_i) if $z(i) = 0$ and $v_i \succ u$ if $z(i) = -1$. Note that fixing the manipulator's vote on all pairwise comparisons in $\mathcal{A}$ involving u fixes the pBorda score of u. Similarly, define the *safety-set* of a vertex $u \in I_{T,E_T}$ as the set of all vectors $z \in \{-1, 0, 1\}^{|T|}$ with respect to which u is *safe*. We say that a vertex $u \in I_{T,E_T}$ *sees* a vote $z \in \{-1,0,1\}^{|T|}$ if z is the restriction of the manipulator's vote to the pairwise comparisons in $\mathcal{A}$ involving u.

Given an equivalence class I_{T,E_T} and the safety-set for each $u \in I_{T,E_T}$, define a *safe-subclass* as the set of all vertices in I_{T,E_T} with identical safety-sets. Denote the number of safe-subclasses in I_{T,E_T} by N_{T,E_T}. Thus, $N_{T,E_T} \leq 3^k$.

We now claim that any valid solution to pBorda-Manipulation can be transformed into another (possibly different) solution where all vertices inside a safe-subclass *see* the same vote vector. Indeed, fix a safe-subclass and let z' be the restriction of a valid vote $\succ$ as *seen* by the vertex with the highest excess score in that safe-subclass. An alternate vote can now be constructed as follows: in the original vote $\succ$, replace the vote vector currently *seen* by each vertex inside the given safe-subclass by z', while keeping the rest of the vote unchanged. It is easy to check that the excess score constraints for all vertices continue to remain satisfied in the new vote. Therefore, without loss of generality, all vertices inside a safe-subclass *see* the same vote vector in a valid vote of the manipulator.

Our algorithm takes as input an instance of pBorda-Manipulation, namely $\langle \Pi, i^*, \mathcal{A} \rangle$ and returns a YES/NO output indicating the existence of a valid manipulative vote (along with a valid vote, if one exists). The algorithm starts by *guessing* the manipulator's vote within the vertex cover (call this guess $\succ_{\mathcal{S}}$). There are at most $\binom{k}{2}$ such pairs, hence the total number of choices is at most $3^{O(k^2)}$. For each such guess, we obtain a new instance of pBorda-Manipulation, namely $\langle \Pi', i^*, \mathcal{A}' \rangle$ where Π' is a voting profile representing the original votes of the non-manipulators combined with the manipulator's vote $\succ_{\mathcal{S}}$ over the vertex cover and $\mathcal{A}'$ represents the restriction of the graph $\mathcal{A}$ to the bipartite subgraph $\mathcal{S} \times \mathcal{I}$. The algorithm now uses ILP to solve this new problem for each equivalence class in parallel, and checks if the combined vote constitutes a valid solution.

Formulating the ILP: We now describe the variables and constraints for the ILP.

Variables: For each subset $T \subseteq S$, each score vector $E_T = \langle (\alpha_1, \beta_1), \ldots, (\alpha_t, \beta_t) \rangle$, each $1 \leq p \leq 3^{|T|}$ and each $1 \leq q \leq N_{T,E_T}$, define a binary variable $Z_{T,E_T,p,q} \in \{0,1\}$. Here $Z_{T,E_T,p,q} = 1$ (respectively 0) indicates that given T, E_T and the induced equivalence class I_{T,E_T}, the safe-subclass indexed by q *sees* (respectively *does not see*) the vote vector indexed by p. Thus, there are at most $2^k \cdot d^k \cdot 3^{2k}$ variables overall. In other words, the number of variables depends only on the parameters d and k.

Constraints: Our ILP has three types of constraints:

(i) *Sanity constraints*:

(a) $Z_{T,E_T,p,q} \in \{0,1\}$ for all $T,\ E_T,\ p$ and q.

(b) for every T, E_T and q, $\sum_p Z_{T,E_T,p,q} = 1$ (i.e. each safe-subclass *sees* exactly one vote vector).

(ii) *Excess score constraints for the vertex cover*: for each vertex $v_i \in S$

$$\begin{aligned}\textstyle\sum_{T \in T_v} \sum_{E_T} \sum_q \sum_p & Z_{T,E_T,p,q} \cdot \mathbf{1}(p_i = 0) \cdot \tfrac{\alpha_i}{\alpha_i + \beta_i} \cdot |q| \\ &+ Z_{T,E_T,p,q} \cdot \mathbf{1}(p_i = -1) \cdot \tfrac{\alpha_i + 1}{\alpha_i + \beta_i + 1} \cdot |q| \\ &+ Z_{T,E_T,p,q} \cdot \mathbf{1}(p_i = +1) \cdot \tfrac{\alpha_i}{\alpha_i + \beta_i + 1} \cdot |q| \leq s^*\end{aligned}$$

where $T_v = \{T \subseteq S \,|\, v \in T\}$ and $|q|$ represents the cardinality of the safe-subclass q. The latter can be efficiently precomputed.

(iii) *Excess score constraints for the independent set*: for all T, E_T, p and q

$$Z_{T,E_T,p,q} \leq Z^{\text{safe}}_{T,E_T,p,q}$$

where $Z^{\text{safe}}_{T,E_T,p,q} \in \{0,1\}$ is a (precomputed) binary indicator specifying whether, given T, E_T and the induced equivalence class I_{T,E_T}, the vector p belongs to the safety-set of (any vertex in) the safe-subclass q.

The theorem now follows since ILP feasibility is FPT when parameterized by the number of variables [30], which, as remarked earlier, is a function of d and k alone. □

Remark 3.3. *The proof techniques used in our algorithmic results (Theorems 3.4 and 3.5) can be readily applied to* S-Elimination *to recover the corresponding results in [29].*

4. RELATED WORK

Parameterized complexity analysis has proven extremely useful in scrutinizing the computational behavior of a variety of problems in computational social choice, namely *winner-determination* [22, 31, 24, 32], *manipulation* [23, 33, 34, 26, 27, 35], *bribery* [36, 37, 38], *possible and necessary winner problems* [39, 40, 41], etc. We refer the reader to [42, 43] for detailed surveys on this topic.

Among the studies on the parameterized complexity of manipulation of standard voting rules, our work shares the spirit of [34, 35] where parameterization of the preference domain (in their case, in terms of closeness to single-peakedness) was used to show special-case tractability results. Specifically, [34] showed that unweighted Borda manipulation with two manipulators is efficiently solvable over the domain of single-peaked preferences, although the problem is known to be NP-complete over the unrestricted domain [44, 33].[3] This result was later generalized in [35] where the manipulation problems for Borda and Copeland$^\alpha$ rules by two manipulators were shown to be FPT in the parameter *single-peaked width* (which measures the distance of a preference profile from single-peakedness).

5. CONCLUDING REMARKS

We studied the problem of manipulation in the model of pairwise preferences and gave a complete classification of the parameterized complexity of manipulating the pairwise Borda rule in terms of various natural parameters relating to the action space. This involved the introduction of *diversity* as a parameter, which we demonstrated to be useful from an algorithmic perspective.

Our work opens up two very natural directions for future work. First, the parameterized complexity of pBorda-Manipulation for other settings of the parameter `pref-type` remains to be analyzed. Second, it would be interesting to compare the parameterized behavior of pBorda rule with that of other pairwise voting rules like Copeland$^\alpha$ [17] (the classical complexity landscape for this family of rules was described in [10]), PageRank [45], HodgeRank [46], Ranked Pairs, Schulze's rule [47] etc.

[3] Recall that the problem of unweighted Borda manipulation by a single manipulator was shown to be efficiently solvable over the domain of rankings in [4].

6. ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their helpful comments. NM was supported by a DST INSPIRE Faculty grant [IFA12-ENG-31]. RV thanks Google and Microsoft Research for travel grants.

REFERENCES

[1] Kenneth J Arrow, Amartya Sen, and Kotaro Suzumura. *Handbook of Social Choice & Welfare*, volume 2. Elsevier, 2010.

[2] A. Gibbard. Manipulation of Voting Schemes: a General Result. *Econometrica: Journal of the Econometric Society*, pages 587–601, 1973.

[3] M.A. Satterthwaite. Strategy-proofness and Arrow's Conditions: Existence and Correspondence Theorems for Voting Procedures and Social Welfare Functions. *Journal of Economic Theory (JET)*, 10(2):187–217, 1975.

[4] John J. Bartholdi III, Craig A. Tovey, and Michael A. Trick. The computational difficulty of manipulating an election. *Social Choice and Welfare*, 6(3):227–241, 1989.

[5] Piotr Faliszewski and Ariel D. Procaccia. AI's war on manipulation: Are we winning? *AI MAGAZINE*, 2010.

[6] Nir Ailon. Aggregation of partial rankings, p-ratings and top-m lists. *Algorithmica*, 57(2):284–300, 2010.

[7] Nina Narodytska and Toby Walsh. The Computational Impact of Partial Votes on Strategic Voting. In *Proceedings of the 21st European Conference on Artificial Intel ligence (ECAI)*, pages 657–662, 2014.

[8] Vijay Menon and Kate Larson. Complexity of Manipulation in Elections with Top-truncated Ballots. *arXiv preprint arXiv:1505.05900*, 2015.

[9] Maria Silvia Pini, Francesca Rossi, Kristen Brent Venable, and Toby Walsh. Aggregating Partially Ordered Preferences. *Journal of Logic and Computation*, 19(3):475–502, June 2009.

[10] Rohit Vaish, Neeldhara Misra, Shivani Agarwal, and Avrim Blum. On the computational hardness of manipulating pairwise voting rules. In *Proceedings of the Fifteenth International Conference on Autonomous Agents and Multiagent Systems (to appear)*, 2016.

[11] R. G. Downey and M. R. Fellows. *Parameterized complexity*. Springer-Verlag, New York, 1999.

[12] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.

[13] R. Niedermeier. *Invitation to fixed-parameter algorithms*, volume 31 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2006.

[14] Marek Cygan, Fedor V Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[15] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[16] Arun Rajkumar and Shivani Agarwal. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *Proceedings of the 31st International Conference on Machine Learning*, pages 118–126, 2014.

[17] Piotr Faliszewski, Edith Hemaspaandra, and Henning Schnoor. Copeland voting: Ties matter. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 983–990. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[18] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[19] Benjamin L Schwartz. Possible winners in partially completed tournaments. *SIAM Review*, 8(3):302–308, 1966.

[20] Walter Kern and Daniël Paulusma. The Computational Complexity of the Elimination Problem in Generalized Sports Competitions. *Discrete Optimization*, 1(2):205–214, November 2004.

[21] Michael Dom, Daniel Lokshtanov, Saket Saurabh, and Yngve Villanger. Capacitated domination and covering: A parameterized perspective. In *Parameterized and Exact Computation*, pages 78–90. Springer, 2008.

[22] John J. Bartholdi III, Craig A. Tovey, and Michael A. Trick. Voting Schemes for which It Can Be Difficult to Tell Who Won the Election. *Social Choice and Welfare*, 6(2):157–165, 1989.

[23] Vincent Conitzer, Tuomas Sandholm, and Jérôme Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3):14, 2007.

[24] Nadja Betzler, Michael R Fellows, Jiong Guo, Rolf Niedermeier, and Frances A Rosamond. Fixed-parameter algorithms for Kemeny rankings. *Theoretical Computer Science*, 410(45):4554–4570, 2009.

[25] Piotr Faliszewski, Edith Hemaspaandra, Lane A Hemaspaandra, and Jörg Rothe. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, pages 275–341, 2009.

[26] Yongjie Yang. Election Attacks with Few Candidates. In *ECAI*, pages 1131–1132, 2014.

[27] Lirong Xia. Fixed-parameter tractability of integer generalized scoring rules. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, pages 1599–1600. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[28] Jiong Guo, Falk Hüffner, and Rolf Niedermeier. A Structural View on Parameterizing Problems: Distance from Triviality. In *Parameterized and Exact Computation*, pages 162–173. Springer, 2004.

[29] Katarina Cechlárová, Eva Potpinková, and Ildikó Schlotter. Refining the complexity of the sports elimination problem. *Discrete Applied Mathematics*, 199:172 – 186, 2016.

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

[31] Nadja Betzler, Jiong Guo, and Rolf Niedermeier. Parameterized computational complexity of Dodgson and Young elections. *Information and Computation*, 208(2):165–177, 2010.

[32] Michael R Fellows, Bart MP Jansen, Daniel Lokshtanov, Frances A Rosamond, and Saket Saurabh. Determining the Winner of a Dodgson Election is Hard. In *FSTTCS*, pages 459–468, 2010.

[33] Nadja Betzler, Rolf Niedermeier, and Gerhard J Woeginger. Unweighted Coalitional Manipulation under the Borda Rule Is NP-Hard. In *IJCAI*, volume 11, pages 55–60. Citeseer, 2011.

[34] Yongjie Yang and Jiong Guo. Exact algorithms for weighted and unweighted Borda manipulation problems. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, pages 1327–1328. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[35] Yongjie Yang. Manipulation with bounded single-peaked width: A parameterized study. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 77–85. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

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

[37] Jianxin Wang, Min Yang, Jiong Guo, Qilong Feng, and Jianer Chen. Parameterized complexity of control and bribery for d-approval elections. In *Combinatorial Optimization and Applications*, pages 260–271. Springer, 2013.

[38] Robert Bredereck, Jiehua Chen, Piotr Faliszewski, André Nichterlein, and Rolf Niedermeier. Prices Matter for the Parameterized Complexity of Shift Bribery. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[39] Nadja Betzler, Susanne Hemmann, and Rolf Niedermeier. A Multivariate Complexity Analysis of Determining Possible Winners Given Incomplete Votes. In *IJCAI*, volume 9, pages 53–58, 2009.

[40] Lirong Xia and Vincent Conitzer. Determining Possible and Necessary Winners under Common Voting Rules Given Partial Orders. *Journal of Artificial Intelligence Research*, pages 25–67, 2011.

[41] Dorothea Baumeister, Piotr Faliszewski, Jérôme Lang, and Jörg Rothe. Campaigns for lazy voters: Truncated ballots. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 577–584. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[42] Claudia Lindner and Jörg Rothe. Fixed-parameter tractability and parameterized complexity, applied to problems from computational social choice. In *Mathematical Programming Glossary*, 2008.

[43] Nadja Betzler, Robert Bredereck, Jiehua Chen, and Rolf Niedermeier. Studies in Computational Aspects of Voting: a Parameterized Complexity Perspective. In *The Multivariate Algorithmic Revolution and Beyond*, pages 318–363. Springer-Verlag, 2012.

[44] Jessica Davies, George Katsirelos, Nina Narodytska, Toby Walsh, and Lirong Xia. Complexity of and algorithms for the manipulation of Borda, Nanson's and Baldwin's voting rules. *Artificial Intelligence*, 217:20–42, 2014.

[45] Sergey Brin and Lawrence Page. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks*, 56(18):3825–3833, 2012.

[46] Xiaoye Jiang, Lek-Heng Lim, Yuan Yao, and Yinyu Ye. Statistical ranking and combinatorial Hodge theory. *Mathematical Programming*, 127(1):203–244, March 2011.

[47] David C Parkes and Lirong Xia. A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12)*. American Association for Artificial Intelligence, 2012.