Being Caught Between a Rock and a Hard Place in an Election—Voter Deterrence by Deletion of Candidates

Britta Dorn[1] and Dominikus Krüger[2]

[1] Universität Tübingen, Wilhelm-Schickard-Institut für Informatik, Sand 13, 72076 Tübingen, Germany
[2] Universität Ulm, Institut für Theoretische Informatik, James-Franck-Ring 5 / O27, 89081 Ulm, Germany

Abstract. We introduce a new problem modeling voter deterrence by deletion of candidates in elections: In an election, the removal of certain candidates might deter some of the voters from casting their votes, and the lower turnout then could cause a preferred candidate to win the election. This is a special case of the variant of the CONTROL problem in which an external agent is allowed to delete candidates and votes in order to make his preferred candidate win, and a generalization of the variant where candidates are deleted, but no votes. We initiate a study of the computational complexity of this problem for several voting systems and obtain $\mathcal{NP}$-completeness and $\mathcal{W}[2]$-hardness with respect to the parameter *number of deleted candidates* for most of them.

1 Introduction

Imagine: finally, you have the chance of getting rid of your old mayor, whom you absolutely cannot stand. Luckily, in addition to the normal unscrupulous opponents, the perfect candidate is running for the vote this year. You agree with everything he says and therefore you are even looking forward to Election Day. But suddenly the word is spread that he has withdrawn his candidacy. Again, you are feeling caught between a rock and a hard place. Does it make any sense to go to the polls if you only have a choice between the lesser of two evils?

Low voter turnouts caused by scenarios such as the one in the above example may lead to modified outcomes of an election. This is reminiscent of a family of problems which have been studied extensively in the computational social choice literature recently, the CONTROL problems [1–5] where an external agent can change the outcome of an election by adding or deleting candidates and/or voters, respectively. In particular, in the setting of constructive control by deleting candidates, the agent can prevent candidates from running for office, which causes other candidates to rise in ranking for certain voters. This may ultimately result in the external agent's preferred candidate winning the election.

In real life, this process is a little bit more complicated and control of an election can occur in a more entangled way: As in our introductory example, if some candidates do not stand for election, then certain voters will not even take part in the election because they feel that there is nothing interesting to decide or no relevant candidate to vote for. The lower turnout could have consequences for the remaining candidates: the winner of the election under normal conditions might lose points because of the lower polling after the deletion of certain candidates, and this can produce a different winner. Hence, by *deterring* the voters by means of deleting their favorite candidates, one might prevent them from casting their votes and therefore change the outcome of the election. Therefore, we call this phenomenon *voter deterrence*.

This situation can be observed in the primaries in US elections or in mayoral elections, where mayors often are elected with single-digit turnout, sometimes caused by the withdrawal of candidacy of one or several alternatives in the run-up.

As to our knowledge, this problem has not yet been considered from a computational point of view. In this paper, we want to initiate the study of the corresponding decision problem VOTER DETERRENCE defined below. We mainly consider the case where voters are easily deterred: As soon as their most preferred candidate does not participate in the election, they refrain from the election. This is what we denote as 1-VOTER DETERRENCE, but clearly, one can also consider x-VOTER DETERRENCE, where a voter only refuses to cast his vote if his top x candidates are removed. Surprisingly, it turns out that 1-VOTER DETERRENCE is already computationally hard for several voting systems, even for Veto.

This paper is organized as follows. After introducing notation and defining the decision problem x-VOTER DETERRENCE in Section 2, we investigate the complexity of this problem for the case of $x = 1$ for the voting systems Plurality (for which it turns out to be solvable in polynomial time, but with $x = 2$ it is $\mathcal{NP}$-complete), Veto, 2-approval, Borda, Maximin, Bucklin, Fallback Voting, and Copeland (for all of which the problem turns out to be $\mathcal{NP}$-complete). As a corollary, we can show that the hard problems are also $\mathcal{W}[2]$-hard with respect to the solution size, i.e., with respect to the parameter *number of deleted candidates*, which means that they remain hard even if only few candidates have to be deleted to make the preferred candidate win. This is stated in Section 4 together with a short discussion of the complexity with respect to the parameter *number of candidates*. We conclude with a discussion of open problems and further directions that might be interesting for future investigations.

2 Preliminaries

Elections. An *election* is a pair $E = (C, V)$ consisting of a *candidate* set $C = \{c_1, \ldots, c_m\}$ and a multiset $V = \{v_1, \ldots, v_n\}$ of *votes* or *voters*, each of them a linear order over C, i.e., a transitive, antisymmetric, and total relation over the candidates in C, which we denote by $\succ$. A *voting system* maps (C, V) to a set $W \subseteq C$ called the *winners* of the election. All our results are given for the *unique winner case*, where W consists of a single candidate.

We will consider several voting systems. Each of them is shortly described in the corresponding subsection. Most of them are *positional* scoring protocols, which are defined by a vector of integers $\boldsymbol{\alpha} = \langle \alpha_1, \ldots, \alpha_m \rangle$, with m being the number of candidates. For each voter, a candidate receives α_1 points if he is ranked first by the voter, α_2 if he is ranked second, etc. The *score* of a candidate is the total number of points the candidate receives. Normally, whenever candidates receive points in a voting system, the one with the highest score wins.

Voter Deterrence, Control. In an x-Voter Deterrence instance, we consider a fixed natural number x, and we are given an election $E = (C, V)$, a preferred candidate $p \in C$, and a natural number $k \leq |C|$, as well as a voting system. It will always be clear from the context which voting system we are using, so we will not mention it explicitly in the problem description. Let $R \subseteq C$ denote a subset of candidates, and let $V_R \subseteq V$ denote the set of voters who have ranked only candidates from R among the first x ranks in their vote. The task consists in determining a set R of at most k candidates that are removed from C, and who therefore prevent the set of voters V_R from casting their votes, such that p is a winner in the election $\widetilde{E} = (C \setminus R, V \setminus V_R)$. The set R is then called a *solution* to the x-Voter Deterrence instance. The underlying decision problem is the following.

> x-Voter Deterrence
> **Given:** An election $E = (C, V)$, a preferred candidate $p \in C$, $k \in \mathbb{N}$ and a fixed $x \in \mathbb{N}$.
> **Question:** Is there a subset of candidates $R \subseteq C$ with $|R| \leq k$, such that p is the winner in the election $\widetilde{E} = (C \setminus R, V \setminus V_R)$?

x-Voter Deterrence is a special case of one of the many variants of the Control problem [5], where the chair is allowed to delete candidates and votes, which is defined as follows.

> Constructive Control by Deleting Candidates and Votes
> **Given:** An election $E = (C, V)$, a preferred candidate $p \in C$, and $k, l \in \mathbb{N}$.
> **Question:** Is there a subset $C' \subseteq C$ with $|C'| \leq k$, and a subset $V' \subseteq V$ with $|V'| \leq l$, such that p is a winner in the election $\widetilde{E} = (C \setminus C', V \setminus V')$?

Note that in the Voter Deterrence problem, the deleted candidates and votes are coupled, which is not necessarily the case in the above Control problem. If we set $x = m$, we obtain Constructive Control by Deleting Candidates, which is the above Constructive Control by Deleting Candidates and Votes problem with $l = 0$. The latter variant hence is a special case of m-Voter Deterrence, implying that the hardness results from [1] carry over to m-Voter Deterrence.

In this paper, we will mainly consider 1-Voter Deterrence, i.e., a voter will refuse to cast his vote if his most preferred candidate does not participate in the election. For the voting system Plurality, we also consider 2-Voter Deterrence, where a voter only refrains from voting if his two top ranked candidates are eliminated from the election.

Parameterized complexity. The computational complexity of a problem is usually studied with respect to the size of the input I of the problem. One can also consider the parameterized complexity [6–8] taking additionally into account the size of a so-called parameter k which is a certain part of the input, such as the number of candidates in an election, or the size of the solution set. A problem is called *fixed-parameter tractable* with respect to a parameter k if it can be solved in $f(k) \cdot |I|^{O(1)}$ time, where f is an arbitrary computable function depending on k only. The corresponding complexity class consisting of all problems that are fixed-parameter tractable with respect to a certain parameter is called $\mathcal{FPT}$.

The first two levels of (presumable) parameterized intractability are captured by the complexity classes $\mathcal{W}[1]$ and $\mathcal{W}[2]$. Proving hardness with respect to these classes can be done using an $\mathcal{FPT}$*-reduction*, which reduces a problem instance (I, k) in $f(k) \cdot |I|^{O(1)}$ time to an instance (I', k') such that (I, k) is a yes-instance if and only if (I', k') is a yes-instance, and k' only depends on k but not on $|I|$, see [6–8].

For all our hardness proofs, we use the $\mathcal{W}[2]$-complete Dominating Set (DS) problem for undirected graphs.

Dominating Set
Given: An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and a nonnegative integer k.
Question: Is there a subset $\mathcal{V}' \subseteq \mathcal{V}$ with $|\mathcal{V}'| \leq k$ such that every vertex $v \in \mathcal{V}$ is contained in $\mathcal{V}'$ or has a neighbor in $\mathcal{V}'$?

Notation in our proofs. In all our reductions from Dominating Set, we will associate the vertices of the given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with candidates of the election $E = (C' \cup I, V)$ to be constructed. For that sake, we use a bijection $g \colon \mathcal{V} \to I$. By $N(v) := \{u \in \mathcal{V} \mid \{u, v\} \in \mathcal{E}\}$, we denote the set of *neighbors* or the *neighborhood* of a vertex $v \in \mathcal{V}$. Analogously, we define the *neighborhood of a candidate* $c_i \in I$ as $N(c_i) = g(N(v_i))$ for $c_i = g(v_i)$, i.e., the set of neighbors of a candidate $c_i \in I$ corresponding to the vertex $v_i \in \mathcal{V}$ is the set of candidates corresponding to the neighborhood of v_i in $\mathcal{G}$. By $\overline{N(v_i)} \subset I$ we denote the set of non-neighbors of v_i, analogously for neighborhoods of candidates.

In our reductions, we usually need one dummy candidate for every $c_i \in C$, these will be denoted by $\hat{c}_i$. All other dummy candidates appearing are marked with a hat as well, usually they are called $\hat{d}$ or similarly. When building the votes in our reductions, we write '$k \parallel a_1 \succ \cdots \succ a_l$' which means that we construct the given vote $a_1 \succ \cdots \succ a_l$ exactly k times.

In our preference lists, we sometimes specify a whole subset of candidates, e.g., $c \succ D$ for a candidate $c \in C$ and a subset of candidates $D \subset C$. This notation means $c \succ d_1 \succ \cdots \succ d_l$ for an arbitrary but fixed order of $D = \{d_1, \ldots, d_l\}$. If we use a set $\overrightarrow{D}$ in a preference list, we mean one specific, fixed (but arbitrary, and unimportant) order of the elements in D, which is reversed if we write $\overleftarrow{D}$. Hence, if $c \succ \overrightarrow{D}$ stands for $c \succ d_1 \succ \cdots \succ d_l$, then $c \succ \overleftarrow{D}$ means $c \succ d_l \succ \cdots \succ d_1$. Finally, whenever we use the notation D_{rest} for a subset of candidates in a vote, we mean the set consisting of those candidates in D that have not been positioned explicitly in this vote.

3 Complexity-theoretic analysis

In this section, we will give several hardness proofs for Voter Deterrence for different voting systems. All our results rely on reductions from the $\mathcal{NP}$-complete problem Dominating Set. We only prove $\mathcal{NP}$-hardness for the different voting systems, but since membership in $\mathcal{NP}$ is always trivially given, $\mathcal{NP}$-completeness follows immediately. For all these reductions we assume that every vertex of the input instance has at least two neighbors, which is achievable by a simple polynomial time preprocessing.

We will give the first of the following reduction proofs in detail. For the remaining reductions, we specify the constructed instances together with further helpful remarks. The proofs that these are indeed equivalent to the Dominating Set-instances are straightforward and can be found in the appendix. In each of them, one obtains a solution to the x-Voter Deterrence-instance by deleting exactly those candidates that correspond to the vertices belonging to a solution of the corresponding Dominating Set-instance, and vice versa.

3.1 Plurality

The *Plurality* protocol is the positional scoring protocol with $\boldsymbol{\alpha} = \langle 1, 0, \ldots, 0 \rangle$ [9].

It is easy to see that 1-Voter Deterrence can be solved efficiently for Plurality. One can simply order the candidates according to their score and if there are more than k candidates ahead of p, this instance is a no-instance. Otherwise p will win after deletion of the candidates that were ranked higher than him, because all the votes which they got a point from are removed. Therefore the following theorem holds.

Theorem 1. 1-Voter Deterrence *is in* $\mathcal{P}$ *for* Plurality*.*

For 2-Voter Deterrence, it is not so easy to see which candidates should be deleted. In fact, the problem is $\mathcal{NP}$-complete.

Theorem 2. 2-Voter Deterrence *is* $\mathcal{NP}$*-complete for* Plurality*.*

Proof. We prove Theorem 2 with an $\mathcal{FPT}$-reduction from Dominating Set. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k \rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$ we need one candidate c_i and one dummy candidate $\hat{c}_i$, as well as the preferred candidate p and his dummy candidate $\hat{p}$, so $C = I \cup D \cup \{p\}$ with $I = \{c_1, \ldots, c_n\}$ and $D = \{\hat{c}_1, \ldots, \hat{c}_n, \hat{p}\}$. For ease of presentation we denote $I \cup \{p\}$ by I^*.
Votes: The votes are built as follows.

$$n \parallel p \succ \hat{p} \succ C_{\text{rest}}, \tag{1}$$

$$\forall c_i \in I:$$

$$|N(c_i)| \parallel c_i \succ \hat{c}_i \succ C_{\text{rest}}, \tag{2}$$

$$\forall c_j \in \overline{N(c_i)} \cup \{p\}:$$

$$1 \parallel c_i \succ c_j \succ C_{\text{rest}}. \tag{3}$$

Note that n votes are built for every candidate c_i. Therefore each candidate in I^* has the score n. The score of a candidate can only be decreased if the corresponding candidate himself is deleted. Note also that the score of every dummy candidate cannot exceed $n-1$.
We will now show that one can make p win the election by deleting up to k candidates if and only if the DS-instance has a solution of size at most k.

"⇒": Let S be a given solution to the DS-instance. Then $R = g(S)$ is a solution to the corresponding 2-Voter Deterrence-instance. Since S is a dominating set, every candidate in I will be at least once in the neighborhood of a candidate $c_i \in R$ or be a candidate in R himself. Therefore p is the only candidate who gains an additional point from every deleted candidate $c_x \in R$ from the vote built by (3) and will therefore be the unique winner.

"⇐": Let R be a given solution to a 2-Voter Deterrence-instance. Note that every candidate in I^* has the original score n. These scores can be increased if the corresponding candidate himself is not deleted. If p wins, then with the same argument as before, we see that every candidate $c_x \in I$ either must be a member of the set R, or must not appear as c_j on the second position of the votes built by (3) for at least one candidate of R, hence must be in the neighborhood of R. Therefore $S = g^{-1}(R)$ is a solution to the equivalent DS-instance. □

3.2 Veto

The positional scoring protocol with $\boldsymbol{\alpha} = \langle 1, 1, \ldots, 1, 0 \rangle$ is called *Veto* [9].

Theorem 3. 1-Voter Deterrence *is $\mathcal{NP}$-complete for* Veto.

We show Theorem 3 with an $\mathcal{FPT}$-reduction from Dominating Set. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k \rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$ we need one candidate c_i, as well as the preferred candidate p and $k+1$ dummy candidates, so $C = I \cup D \cup \{p\}$ with $I = \{c_1, \ldots, c_n\}$ and $D = \{\hat{d}_1, \ldots, \hat{d}_{k+1}\}$. For ease of presentation we denote $I \cup \{p\}$ by I^*.
Votes: The votes are built as follows.

$$\forall c_i \in I:$$
$$\forall c_j \in I^* \setminus (N(c_i) \cup \{c_i\}):$$
$$1 \parallel c_i \succ C_{\text{rest}} \succ D \succ c_j, \quad (1)$$
$$\forall c_j \in N(c_i) \cup \{c_i\}:$$
$$1 \parallel p \succ I_{\text{rest}} \succ D \succ c_j, \quad (2)$$
$$\forall \hat{d}_j \in D:$$
$$2 \parallel p \succ I \succ D_{\text{rest}} \succ \hat{d}_j. \quad (3)$$

Note that every vote built by (2) and (3) can only be removed by deleting the candidate p, who should win the election. Therefore these votes will not be removed. Note also that for each set of votes constructed for a candidate $c_i \in I$, every candidate in $C \setminus D$ takes the last position in one of theses votes, hence the score of every such candidate is the same. In contrast, the dummy candidates cannot win the election at all, due to the fact that they are on the last position of the constructed votes twice as often as the other candidates.

3.3 2-approval

The *2-approval* protocol is the positional scoring protocol with the scoring vector $\boldsymbol{\alpha} = \langle 1, 1, 0, \ldots, 0 \rangle$ [9].

Theorem 4. 1-VOTER DETERRENCE *is $\mathcal{NP}$-complete for* 2-approval.

We show Theorem 4 by an $\mathcal{FPT}$-reduction from DOMINATING SET. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k \rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$, we create one candidate c_i and one additional dummy candidate $\hat{c}_i$, finally we need the preferred candidate p. So with $I = \{c_1, \ldots, c_n\}$ and $D = \{\hat{c}_1, \ldots, \hat{c}_n\}$, the candidates are $C = I \cup D \cup \{p\}$.
Votes: The votes are built as follows.

$$\begin{aligned}
&\forall c_i \in I : \\
&\qquad \forall c_j \in N(c_i) : \\
&\qquad\qquad 1 \parallel c_i \succ c_j \succ \hat{c}_j \succ C_{\text{rest}} \succ p, && (1)\\
&\qquad \forall c_j \in I \setminus (N(c_i) \cup \{c_i\}) : \\
&\qquad\qquad 1 \parallel \hat{c}_i \succ c_j \succ \hat{c}_j \succ C_{\text{rest}} \succ p, && (2)\\
&\qquad 2 \parallel \hat{c}_i \succ p \succ C_{\text{rest}}, && (3)\\
&\qquad n - |N(c_i)| \parallel c_i \succ \hat{c}_i \succ C_{\text{rest}} \succ p. && (4)
\end{aligned}$$

Without any candidate deleted, all $c_i \in I$ and p have the same score of $2n$, while the dummy candidates $\hat{c}_j \in D$ have a score less than $2n$. Note that one decreases p's score by deleting a dummy candidate, because a deletion of this kind results in losing a vote built in (3). Therefore one has to delete candidates in I to help p in winning.

3.4 Borda

The positional scoring protocol with $\boldsymbol{\alpha} = \langle m-1, m-2, \ldots, 0 \rangle$ is called *Borda* [9].

Theorem 5. 1-VOTER DETERRENCE *is $\mathcal{NP}$-complete for* Borda.

We show Theorem 5 by an $\mathcal{FPT}$-reduction from DOMINATING SET. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k \rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$ we create one candidate c_i and one dummy candidate $\hat{c}_i$, finally we need the preferred candidate p. So the candidates are $C = I \cup D \cup \{p\}$ with $I = \{c_1, \ldots, c_n\}$ and $D = \{\hat{c}_1, \ldots, \hat{c}_n\}$. For ease of

presentation, we denote $I \cup \{p\}$ by I^*.
Votes: The votes are built as follows.

$$\forall c_i \in I:$$

$$\forall c_j \in N(c_i):$$

$$1 \parallel c_i \succ \overrightarrow{I^*}_{\text{rest}} \succ c_j \succ \hat{c}_j \succ \overrightarrow{D}_{\text{rest}} \succ \hat{c}_i, \tag{1}$$

$$1 \parallel c_i \succ c_j \succ \hat{c}_j \succ \overleftarrow{I^*}_{\text{rest}} \succ \overleftarrow{D}_{\text{rest}} \succ \hat{c}_i, \tag{2}$$

$$1 \parallel \hat{c}_i \succ \hat{c}_j \succ c_j \succ \overleftarrow{I^*}_{\text{rest}} \succ c_i \succ \overleftarrow{D}_{\text{rest}}, \tag{3}$$

$$1 \parallel \hat{c}_i \succ \overrightarrow{I^*}_{\text{rest}} \succ \hat{c}_j \succ c_j \succ c_i \succ \overrightarrow{D}_{\text{rest}}. \tag{4}$$

Recall that $\overrightarrow{A}$ denotes one specific order of the elements within the set A which is reversed in $\overleftarrow{A}$. Keeping this in mind, it is easy to see that every candidate in I^* has the same score within one gadget constructed by the four votes built by (1) to (4) for one c_j, while the dummy candidates all have a lower score. Note that the deletion of any candidate will decrease the score of every other candidate. Therefore the scores of the candidates in I have to be decreased more than the one of p, whereas the scores of the candidates in I^* can never be brought below the score of any candidate in D.

3.5 Maximin

This voting protocol is also known as *Simpson*. For any two distinct candidates i and j, let $N(i,j)$ be the number of voters that prefer i to j. The *maximin score* of i is $\min_{j \neq i} N(i,j)$ [9].

Theorem 6. 1-Voter Deterrence *is $\mathcal{NP}$-complete for* Maximin.

We show Theorem 6 by an $\mathcal{FPT}$-reduction from Dominating Set. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k \rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$ we create one candidate c_i and one dummy candidate $\hat{c}_i$, finally we need the preferred candidate p. So the candidates are $C = I \cup D \cup \{p\}$ with $I = \{c_1, \dots, c_n\}$ and $D = \{\hat{c}_1, \dots, \hat{c}_n\}$.
Votes: The votes are built as follows.

$$\forall c_i \in I:$$

$$1 \parallel c_i \succ \overrightarrow{I}_{\text{rest}} \succ \overrightarrow{N}(c_i) \succ p \succ \overrightarrow{D}_{\text{rest}} \succ \hat{c}_i, \tag{1}$$

$$1 \parallel c_i \succ \overleftarrow{N}(c_i) \succ p \succ \overleftarrow{I}_{\text{rest}} \succ \overleftarrow{D}_{\text{rest}} \succ \hat{c}_i, \tag{2}$$

$$1 \parallel \hat{c}_i \succ \overrightarrow{I}_{\text{rest}} \succ p \succ \overrightarrow{N}(c_i) \succ \overrightarrow{D}_{\text{rest}} \succ c_i, \tag{3}$$

$$1 \parallel \hat{c}_i \succ p \succ \overleftarrow{N}(c_i) \succ \overleftarrow{I}_{\text{rest}} \succ \overleftarrow{D}_{\text{rest}} \succ c_i. \tag{4}$$

Recall that $\overrightarrow{A}$ denotes one specific order of the elements within set A which is reversed in $\overleftarrow{A}$. With this in mind, it is easy to see that every candidate in I has the same score as p, namely $2n$. The dummy candidates are not able to win the election as long as at least one of the candidates in I or p is remaining.

3.6 Bucklin and Fallback Voting

A candidate c's Bucklin score is the smallest number k such that more than half of the votes rank c among the top k candidates. The winner is the candidate that has the smallest Bucklin score [10].

Theorem 7. 1-Voter Deterrence *is $\mathcal{NP}$-complete for* Bucklin.

Note that Bucklin is a special case of *Fallback Voting*, where each voter approves of each candidate, see [11]. We therefore also obtain

Corollary 1. 1-Voter Deterrence *is $\mathcal{NP}$-complete for* Fallback Voting.

We show Theorem 7 by an $\mathcal{FPT}$-reduction from Dominating Set. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k\rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$ we create one candidate c_i and one dummy candidate $\hat{c}_i$. Additionally, we need the preferred candidate p and several dummy candidates. We need $n(n+k)$ *filling* dummies $\hat{f}$, $k(2n+k-1)$ *security* dummies $\hat{s}$, and finally $k-1$ *leading* dummies $\hat{l}$. So the candidates are $C = I \cup D \cup S \cup F \cup L \cup \{p\}$ with $I = \{c_1, \ldots, c_n\}$, $D = \{\hat{c}_1, \ldots, \hat{c}_n\}$, $S = \{\hat{s}_1, \ldots, \hat{s}_{k(2n+k-1)}\}$, $F = \{\hat{f}_1, \ldots, \hat{f}_{n(n+k)}\}$, and $L = \{\hat{l}_1, \ldots, \hat{l}_{k-1}\}$. For ease of presentation, we denote $I \cup \{p\}$ by I^*.
Votes: The votes are built as follows.
$\forall c_i \in I:$

$$\begin{aligned} 1 \parallel c_i &\succ N(c_i) \succ \{\hat{f}_{(i-1)(n+1)+1}, \ldots, \hat{f}_{i(n+1)-|N(c_i)|-1}\} \\ &\succ \{\hat{s}_{(2i-2)(k+1)+1}, \ldots, \hat{s}_{2(i-1)(k+1)}\} \succ C_{\text{rest}} \succ p, \qquad (1) \\ 1 \parallel \hat{c}_i &\succ \overline{N(c_i)} \succ \{\hat{f}_{i(n+1)-|N(c_i)|}, \ldots, \hat{f}_{(i)(n+1)}\} \succ p \\ &\succ \{\hat{s}_{(2i-1)(k+1)+1}, \ldots, \hat{s}_{2i(k+1)}\} \succ C_{\text{rest}}, \qquad (2) \end{aligned}$$

$\forall r \in \{1, \ldots, k-1\}:$ one vote of the form

$$\begin{aligned} 1 \parallel \hat{l}_r &\succ \{\hat{f}_{n(n+1)+(r-1)n+1}, \ldots, \hat{f}_{n(n+1)+in}\} \\ &\succ \{\hat{s}_{2n(k+1)+(r-1)(k+1)+1}, \ldots, \hat{s}_{2n(k+1)+r(k+1)}\} \succ C_{\text{rest}} \succ p. \qquad (3) \end{aligned}$$

Note that every candidate in I^* occurs within the first $n+2$ positions in the votes built by (1) and (2) for every candidate $c_i \in I$ exactly once. Therefore p is not the unique winner without modification. Note also that deleting some of the dummy candidates is not helping p, as they all appear just once within the first $n+2$ positions. Because of the security dummies, no candidate in I^* can move up to one of the first $n+2$ positions, if he has not been there before. After the deletion of k candidates, up to k votes can be removed—note that every removed vote has to be built by (1) or (3) if p wins the election with this deletion.

3.7 Copeland

For any two distinct candidates i and j, let again $N(i,j)$ be the number of voters that prefer i to j, and let $C(i,j) = +1$ if $N(i,j) > N(j,i)$, $C(i,j) = 0$ if

$N(i,j) = N(j,i)$, and $C(i,j) = -1$ if $N(i,j) < N(j,i)$. The *Copeland score* of candidate i is $\sum_{j \neq i} C(i,j)$ [9].

Theorem 8. 1-Voter Deterrence *is $\mathcal{NP}$-complete for the voting system* Copeland.

We show Theorem 8 by an $\mathcal{FPT}$-reduction from Dominating Set. Let $\langle \mathcal{G} = (\mathcal{V}, \mathcal{E}), k \rangle$ be an instance of DS.
Candidates: For every vertex $v_i \in \mathcal{V}$ we create one candidate c_i and one dummy candidate $\hat{c}_i$. Additionally we need the preferred candidate p, one *thievish* candidate $\hat{t}$ and furthermore n *filling* dummy candidates. So the candidates are $C = I \cup D \cup F \cup \{\hat{t}, p\}$ with $I = \{c_1, \dots, c_n\}$, $D = \{\hat{c}_1, \dots, \hat{c}_n\}$, and $F = \{\hat{f}_1, \dots, \hat{f}_n\}$.
Votes: The votes are built as follows.

$$\forall c_i \in I:$$

$$1 \parallel c_i \succ \overrightarrow{N}(c_i) \succ \hat{t} \succ \overrightarrow{I}_{\text{rest}} \succ p \succ \overrightarrow{F} \succ \overrightarrow{D}_{\text{rest}} \succ \hat{c}_i, \tag{1}$$

$$1 \parallel c_i \succ p \succ \overleftarrow{I}_{\text{rest}} \succ \overleftarrow{N}(c_i) \succ \overleftarrow{F} \succ \hat{t} \succ \overleftarrow{D}_{\text{rest}} \succ \hat{c}_i, \tag{2}$$

$$1 \parallel \hat{c}_i \succ \hat{t} \succ \overrightarrow{N}(c_i) \succ \overrightarrow{I}_{\text{rest}} \succ p \succ \overrightarrow{F} \succ c_i \succ \overrightarrow{D}_{\text{rest}}, \tag{3}$$

$$1 \parallel \hat{c}_i \succ p \succ \overleftarrow{I}_{\text{rest}} \succ \overleftarrow{F} \succ \hat{t} \succ \overleftarrow{N}(c_i) \succ c_i \succ \overleftarrow{D}_{\text{rest}}. \tag{4}$$

These n gadgets (consisting of the above 4 votes) cause that the candidates have different scores. Note that the candidates of each set are always tying with the other candidates in their set, since every gadget has two votes with one specific order of the members and another two of the reversed order. Since candidates in D are losing every pairwise election against all other candidates, they have a score of $-(2n+2)$. The candidates in F are just winning against the candidates in D and are tied against $\hat{t}$ and therefore have a score of -1. Since the candidates in I and p are on a par with $\hat{t}$, this gives them a score of $2n$ and $\hat{t}$ a score of n. Note that if there exists a deletion of k candidates which makes p win the election, there also exists a deletion of up to k candidates in I doing so. The main idea here is that the thievish candidate can steal exactly one point from every candidate in I by winning the pairwise election between them due to the deleted candidate and thereby removed votes. Since $\hat{t}$ starts with a score of n, this will only bring him to a score of $2n - k$ with k deleted candidates. Therefore he cannot get a higher score than p initially had.

4 Parameterized complexity-theoretic analysis

In this section, we shortly take a closer look at the parameterized complexity of Voter Deterrence for the previously considered voting systems.

Since all the $\mathcal{NP}$-hardness proofs of the previous section are based on $\mathcal{FPT}$-reductions from Dominating Set, we immediately obtain

Corollary 2. 1-Voter Deterrence *is $\mathcal{W}[2]$-hard for* Copeland*,* Veto*,* Borda*,* 2-approval*,* Maximin*,* Bucklin*, and* Fallback Voting*, and* 2-Voter Deterrence *is $\mathcal{W}[2]$-hard for* Plurality*, all with respect to the parameter* number of deleted candidates*.*

In contrast, considering a different parameter, one easily obtains the following tractability result.

Theorem 9. *The problem x-*Voter Deterrence *is in $\mathcal{FPT}$ with respect to the parameter* number of candidates *for all voting systems having a polynomial time winner determination.*

It is easy to see that Theorem 9 holds: An algorithm trying out every combination of candidates to delete has an $\mathcal{FPT}$-running time $\mathcal{O}(m^k \cdot n \cdot m \cdot T_{\text{poly}})$, where m is the number of candidates, n the number of votes, $k \leq m$ is the number of allowed deletions, and T_{poly} is the polynomial running time of the winner determination in the specific voting system.

5 Conclusion

We have initiated the study of a voting problem that takes into account correlations that appear in real life, but which has not been considered from a computational point of view so far. We obtained $\mathcal{NP}$-completeness and $\mathcal{W}[2]$-hardness for most voting systems we considered. However, this is just the beginning, and it would be interesting to obtain results for other voting systems such as k-approval or scoring rules in general. Also, we have concentrated on the case of 1-Voter Deterrence and so far have investigated 2-Voter Deterrence for Plurality only.

One could also look at the *destructive* variant of the problem in which an external agent wants to prevent a hated candidate from winning the election, see, e.g., [2] for a discussion for the Control problem.

We have also investigated our problem from the point of view of parameterized complexity. It would be interesting to consider different parameters, such as the number of votes, or even a combination of several parameters (see [12]), to determine the complexity of the problem in a more fine-grained way. This approach seems especially worthwile because Voter Deterrence, like other ways of manipulating the outcome of an election, is a problem for which NP-hardness results promise some kind of resistance against this dishonest behavior. Parameterized complexity helps to keep up this resistance or to show its failure for cases where certain parts of the input are small, and thus it provides a more robust notion of hardness. See, e.g., [11, 13–16], and the recent survey [17].

However, one should keep in mind that combinatorial hardness is a worst case concept, so it would clearly be interesting to consider the average case complexity of the problem or to investigate the structure of naturally appearing instances. E.g., when the voters have *single peaked preferences*, many problems become easy [18]. Research in this direction is becoming more and more popular in the computational social choice community, see for example [18–20].

Acknowledgments. We thank Oliver Gableske for the fruitful discussion which initiated our study of Voter Deterrence, and the referees of COMSOC 2012 and SOFSEM 2013 whose constructive feedback helped to improve this work.

References

1. Bartholdi, J., Tovey, C., Trick, M., et al.: How Hard is it to Control an Election? Math. Comput. Mod. **16**(8-9) (1992) 27–40
2. Hemaspaandra, E., Hemaspaandra, L.A., Rothe, J.: Anyone but him: The complexity of precluding an alternative. Artif. Intell. **171**(5-6) (2007) 255–285
3. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L., Rothe, J.: Llull and Copeland voting computationally resist bribery and constructive control. J. Artif. Intell. Res. **35**(1) (2009) 275–341
4. Erdélyi, G., Rothe, J.: Control complexity in fallback voting. In: Proc. of 16th CATS. (2010) 39–48
5. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L.: Multimode control attacks on elections. J. Artif. Intell. Res. **40** (2011) 305–351
6. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Monographs in Computer Science. Springer, New York (1999)
7. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. An EATCS Series. Springer, New York (2006)
8. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press (2006)
9. Conitzer, V., Lang, J., Sandholm, T.: How many candidates are needed to make elections hard to manipulate? In: Proc. of 9th TARK, ACM (2003) 201–214
10. Xia, L., Zuckerman, M., Procaccia, A.D., Conitzer, V., Rosenschein, J.S.: Complexity of unweighted coalitional manipulation under some common voting rules. In: Proc. 21st IJCAI. (2009) 348–353
11. Erdélyi, G., Fellows, M.: Parameterized control complexity in bucklin voting and in fallback voting. Proc. of 3rd COMSOC (2010) 163–174
12. Niedermeier, R.: Reflections on multivariate algorithmics and problem parameterization. In: Proc. of 27th STACS. (2010) 17–32
13. Betzler, N., Uhlmann, J.: Parameterized complexity of candidate control in elections and related digraph problems. Theor. Comput. Sci. **410**(52) (2009) 5425–5442
14. Christian, R., Fellows, M., Rosamond, F., Slinko, A.: On complexity of lobbying in multiple referenda. Rev. Econ. Design **11**(3) (2007) 217–224
15. Dorn, B., Schlotter, I.: Multivariate complexity analysis of swap bribery. Algorithmica **64**(1) (2012) 126–151
16. Bredereck, R., Chen, J., Hartung, S., Niedermeier, R., Suchỳ, O., Kratsch, S.: A Multivariate Complexity Analysis of Lobbying in Multiple Referenda. In: Proc. of 26th AAAI. 1292–1298
17. Betzler, N., Bredereck, R., Chen, J., Niedermeier, R.: Studies in Computational Aspects of Voting—a Parameterized Complexity Perspective. In Bodlaender et al., H., ed.: Fellows Festschrift, LNCS 7370. Springer, Heidelberg (2012) 318–363
18. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L.A., Rothe, J.: The shield that never was: Societies with single-peaked preferences are more open to manipulation and control. Inf. Comput. **209**(2) (2011) 89–107
19. Friedgut, E., Kalai, G., Nisan, N.: Elections can be manipulated often. In: Proc. of 49th FOCS. (2008) 243–249
20. Faliszewski, P., Procaccia, A.D.: AI's War on Manipulation: Are We Winning? AI Mag. **31**(4) (2010) 53–64