ECAI 2014
T. Schaub et al. (Eds.)

doi:10.3233/978-1-61499-419-0-579

How Hard is Control in Single-Crossing Elections?

Krzysztof Magiera [1] and **Piotr Faliszewski** [2]

Abstract. Election control problems model situations where some entity (traditionally called the election chair) wants to ensure some agent's victory by either adding or deleting candidates or voters. The complexity of deciding if such control actions can be successful is well-studied for many typical voting rules and, usually, such control problems are NP-complete. However, Faliszewski et al. [16] have shown that many control problems become polynomial-time solvable when we consider single-peaked elections. In this paper we show that a similar phenomenon applies to the case of single-crossing elections. Specifically, we consider the complexity of control by adding/deleting candidates/voters under Plurality and Condorcet voting. For each of these control types and each of the rules, we show that if the control type is NP-complete in general, it becomes polynomial-time solvable for single-crossing elections.

1 Introduction

The goal of this paper is to study the complexity of election control, for the case where voters' preferences are single-crossing. While in general, without any restrictions on voters' preferences, the complexity of control problems tends to be high, Faliszewski et al. [16] have shown that it can drop significantly if we assume preferences to be single-peaked. We complement their results by showing the same phenomenon under the single-crossing assumption.

On the intuitive level, election control problems model settings where some entity (traditionally referred to as the *election chair*) is interested in affecting the result of a given election by changing its structure. Typical cases include, e.g., control by deleting candidates or control by adding voters. The former may correspond, e.g., to a situation where an election organizer prevents some candidates from registering, whereas the latter may correspond, e.g., to campaigns aimed at convincing members of the society to cast their votes (of course, a political party that runs such a campaign is most interested in convincing the people that support the party).

Election control problems arise not only in standard political elections, but also (or, perhaps, mostly) in multiagent systems. Ephrati and Rosenschein [10] proposed voting-based solutions for multiagent planning problems, and there control by adding (deleting) candidates can correspond to a particular agent proposing (or arguing for dismissal) of particular variants of the constructed plan. Voting is a natural tool for recommendation systems [17, 20] and there adding/deleting voters can correspond to a situation where a insincere user of a recommendation algorithm tries to manipulate its results by limiting/extending the available preference data. (For example, this insincere user may want to manipulate the results if he or she is trying to influence some other entity's business decisions and these decisions are based on the output of the recommendation system.) Finally, in the well-known scenario where one uses a voting-based meta-search engine for the Web [8] (i.e., an algorithm that treats other web search engines as voters and aggregates their results using voting mechanisms), control by adding/deleting candidtes/voters corresponds to a particular selection of the web pages and search engines to use. (Again, the owner of such a meta-search engine might want to skew its results in some way.)

Election control typically corresponds to negative actions (though, we should stress that this is not always the case) and, so, we would like control to be as hard as possible.[3] Indeed, this idea was at the heart of the research line started by Bartholdi, Tovey, and Trick [1], who introduced the study of the complexity of election control problems and who have shown that candidate control is NP-hard for the Plurality rule and that voter control is hard for the Condorcet rule (we point the reader to Section 2 for detailed definitions). Bartholdi, Tovey, and Trick focused on constructive control, where the manipulating entity tries to ensure some given candidate's victory. Later, Hemaspaandra, Hemaspaandra, and Rothe [18] extended the control model to include *destructive control*, where the goal is to prevent a particular candidate from winning.

Election Control in Restricted Domains. We quickly review some research regarding the complexity of election control at the end of this section. However, briefly put, we have that "for most of the typical voting rules, most of the typical (constructive) control problems are NP-hard." (Note that we do not want to neglect the value of studying control for various voting rules and we think that such research is important; our point is that control problems tend to have high computational complexity.) In light of the previous discussion, we should take it as good news. However, this is so only on the surface. NP-hardness is a worst-case notion and many of the election-control NP-hardness proofs rely on building very contrived elections which are unlikely to appear in real life. Indeed, if in some political elections a voter thinks that a left-wing candidate is the best one, then we would not expect this voter to consider an extreme right-wing candidate to be a second-best. In real life, voters' preference orders have a natural structure and not all preference orders appear.

Social choice theory offers several models that capture such more structured preferences of the voters. Among these models, the two most popular are the single-peaked assumption introduced by Black [2] and the single-crossing assumption, introduced by Mirrlees [22]. In the former, the idea is that there is some global order of the candidates (e.g., modeling the left-to-right political spectrum of opinions) and for each integer k and each voter v, v's k most pre-

[1] AGH University, Krakow, Poland, email: kmagiera@agh.edu.pl
[2] AGH University, Krakow, Poland, email: faliszew@agh.edu.pl

[3] One could even say that it would be ideal if control was not possible at all. This, however, is neither feasible nor desirable. After all, if sufficiently many voters with particular preferences joined the election, we *would* want them to be able to sway the result in their favor. Similarly, if a particularly desirable candidate joined the race, it *would* be natural that he should win. Indeed, for candidate control there even are results suggesting that candidate control is unavoidable in principle [7].

ferred candidates form a consecutive block with respect to this global order. The idea of the single-crossing assumption is that we can order the voters in such a way that the following holds: If a and b are two candidates, then the voters that prefer a to b form a consecutive block on one end of the voter order, and the voters who prefer b to a form a consecutive block on the other end. (In political elections, one could say that the voters are ordered from the extreme-left one to the extreme-right one, and for each two candidates a and b, if a is more "leftist" than b, then the left-wing voters prefer a to b, whereas the right-wing voters would prefer b to a.)

Both single-peaked and single-crossing assumptions are quite convincing (as far as idealized models go, of course). Indeed, as argued by a number of economists, single-crossing profiles arise naturally in many settings (with taxation issues being prominently represented; we point interested readers to the book of Persson and Tabellini [25] for several settings where single-crossing profiles arise). Single-peaked profiles have been studied for much longer and they arise in many natural settings as well (see, e.g., the classic work of Black [2], where single-peaked profiles were introduced). In effect, it is very interesting that Faliszewski et al. [16] and Brandt et al. [3] have shown that if elections are single-peaked, then control by adding/deleting candidates/voters is solvable in polynomial time for both Plurality and Condorcet (and, indeed, for many other voting rules, most notably for Approval). These results suggest that many of the known hardness proofs for control problems rely on building elections that are unlikely to occur in practice. This view was strengthened by Faliszewski et al. [14] who also considered nearly single-peaked elections. In this paper, we provide further evidence that hardness of election control comes from looking at contrived preference profiles by showing that all the control by adding/deleting candidates/voters problems for Plurality and Condorcet for single-crossing elections are solvable in polynomial-time (see Table 1 for a summary).

Related Work. The study of the complexity of control problems was initiated by Bartholdi, Tovey, and Trick [1], who introduced the topic and defined the constructive variants of the problem. Their work was extended by Hemaspaandra, Hemaspaandra, and Rothe [18] to include the destructive cases. Since these two papers, many other researchers studied the complexity of control for various voting rules [15, 24, 11] and in many various settings [21, 19, 13] (the references here are just examples; we also point the reader to the survey of Faliszewski et al. [12]). The study of the complexity of control for restricted domains was initiated by Faliszewski et al. [16], who studied control in single-peaked elections (and who themselves were motivated by results of Walsh [29] and of Conitzer [5]). Brandt et al. [3] and Faliszewski et al. [14] continued this line of work by studying further voting rules and considering nearly single-peaked elections. However, so far there were no papers on control for single-crossing elections and, indeed, there are only very few papers on the complexity of single-crossing elections in general. The few examples include, e.g., the work of Elkind et al. [9] Bredereck et al. [4], Skowron et al. [28], and Cornaz et al. [6].

Organization of the Paper. The paper is organized as follows. First, in Seciton 2 we provide necessary preliminaries regarding election control and the single-crossing assumption. In the next two sections we present our results regarding Plurality and Condorcet elections. We conclude in Section 5.

2 Preliminaries

For each positive integer s, we let $[s]$ denote the set $\{1, ..., s\}$. For a given set S and element e, we define $\mathbf{1}_S(e) = 1$ when $e \in S$ and $\mathbf{1}_S(e) = 0$ otherwise (i.e., $\mathbf{1}_S$ is the characteristic function of S). We assume that the reader is familiar with basic notions of complexity theory, such as the classes P and NP, the notions of NP-hardness and NP-completeness, and basic reducibility types (see, e.g., the textbook of Papadimitriou [23] for more details).

Elections. An election is a pair $E = (C, V)$ where C is a set of candidates and V is a set of voters. Each voter v in V has a *preference order* $\succ_v$, i.e., a linear order over C that ranks all the candidates from the most desirable one to the least desirable one.

Voting Rules. We focus on two voting rules, the Plurality rule and the Condorcet rule, which are defined as follows. Let $E = (C, V)$ be an election. The Plurality score of a candidate c in election E, denoted, $score_E^P(c)$, is the number of voters in V that rank c first. A candidate is a *Plurality winner* if he or she has the highest Plurality score. (In effect, we assume the unique winner model, i.e., we require the winner to be the only candidate with the highest score; however, all our results hold in the nonunique winner model as well.) A candidate is a *Condorcet winner* if he or she defeats every other candidate in a pairwise contest. Formally, for an election $E = (C, V)$, and two candidtes $c, c' \in C$, we write $N_E(c, c')$ to denote the number of voters from V that prefer c to c'. A candidate c is a Condorcet winner if for each candidate $c' \in C \setminus \{c\}$ we have $N_E(c, c') > N_E(c', c)$.

Single-Crossing Elections. Let $E = (C, V)$ be an election, where C is a set of candidates and V is a set of voters. We can alternatively represent the same election by replacing V with an ordered sequence of the same voters, denoted $\hat{V} = (\hat{v}_1, ..., \hat{v}_n)$. Using this notation, we define a single-crossing election as follows [22]:

Definition 1 *Let $E = (C, V)$ be an election and let $\hat{V} = (\hat{v}_1, ..., \hat{v}_n)$ be an ordered sequence of the voters from V. We say that $\hat{V}$ is a* single crossing *order of voters V in election E if for each pair of candidates a, b such that $a \succ_{\hat{v}_1} b$, there exists a value $t_{a,b} \in [n]$ such that $\{i \in [n] \mid a \succ_{\hat{v}_i} b\} = [t_{a,b}]$.*

Definition 2 *An election $E = (C, V)$ is* single-crossing *if there is a single-crossing order of V for E.*

Intuitively, the above definitions say that as we consider the voters in the single-crossing order from one end of the order to the other, the relative ranking of each two candidates changes at most once (that is each pair of candidates "crosses" at most once).

There are efficient polynomial-time algorithms that given an election $E = (C, V)$ check if it is single-crossing and, if so, provide the single-crossing order of the voters [4, 9]. Thus, from now on, we will assume that each single-crossing election comes together with a single-crossing order.

Control problems. We are interested in four variants of election control: control by adding candidates (AC), control by deleting candidates (DC), control by adding voters (AV) and control by deleting voters (DV). For each of these control types we are interested both in constructive control (CC), where the goal is to ensure a given candidate's victory, and in destructive control (DC), where the goal is to prevent some candidate from winning.

Definition 3 *Let R be a voting rule. We are given a set of candidates C, a collection of voters V, a nonnegative integer k and a designated candidate $p \in C$. In control by adding voters (AV) in addition we are given a set W of unregistered voters. Similarly, in control by adding candidates (AC) we are given a set A of unregistered candidates. In constructive variants of control problems we ask whether it is possible for p to became a unique winner under voting rule R in the following way:*

Table 1. The complexity of control problems for Plurality and Condorcet rules, for the unrestricted domain [1, 18], the single-peaked domain [16, 3], and the single-crossing domain (due to the current paper). A dash in an entry means that the given system is *immune* to the type of control in question (i.e., it is impossible to achieve the desired effect by the action this control problem allows; see [1] and [18] for the immunity results).

		Unrestricted		Single-Peaked		Single-Crossing	
Problem		Plurality	Condorcet	Plurality	Condorcet	Plurality	Condorcet
CC	AC	NP-com.	–	P	–	**P**	–
	DC	NP-com.	P	P	P	**P**	**P**
	AV	P	NP-com.	P	P	**P**	**P**
	DV	P	NP-com.	P	P	**P**	**P**
DC	AC	NP-com.	P	P	P	**P**	**P**
	DC	NP-com.	–	P	–	**P**	–
	AV	P	P	P	P	**P**	**P**
	DV	P	P	P	P	**P**	**P**

1. *In constructive control by adding voters (R-CCAV), we ask whether there exists a set $W' \subseteq W$, such that p is the unique winner of R-election $(C, V \cup W')$, where $\|W'\| \leq k$.*
2. *In constructive control by deleting voters (R-CCDV), we ask whether there exists a set $V' \subseteq V$, such that p is the unique winner of R-election $(C, V \setminus V')$, where $\|V'\| \leq k$.*
3. *In constructive control by adding candidates (R-CCAC), we ask whether there exists a set $A' \subseteq A$, such that p is the unique winner of R-election $(C \cup A, V)$, where $\|A'\| \leq k$.*
4. *In constructive control by deleting candidates (R-CCDC), we ask whether there exists a set $C' \subseteq C$, such that p is the unique winner of R-election $(C \setminus C', V)$, where $\|C'\| \leq k$.*

The destructive variants are defined analogously except that we ask whether it is possible to ensure that p is not the unique winner.

When we speak of control problems for the case of single-crossing elections, we mean that the election that contains all the candidates and voters specified in the problem is single-crossing (thus, for example, in CCAC we require that election $(C \cup A, V)$ is single-crossing, and in CCAV we require that election $(C, V \cup W)$ is single-crossing).

3 Plurality Elections

We now present our results for the Plurality rule (see Table 1 for a quick summary). Voter control problems (that is, the problems of constructive and destructive control by adding/deleting voters) are polynomial-time solvable under Plurality [1, 18], so we focus on the candidate control problems, which are known to be NP-complete in the unrestricted case [1, 18].

3.1 Constructive Cases

We show that constructive control by adding/deleting candidates can be solved in polynomial time for Plurality rule, provided that the underlying election is single-crossing. Both these results are based on the same (quite involved) dynamic programming algorithm, that solves a slightly more general problem of control by adding an exact number of candidates. The main idea behind this algorithm is to note that the single-crossing order of the voters implies a certain order in which the candidates should be considered. This is captured in the following lemma.

Lemma 1 *In a single-crossing election $E = (C, V)$, if $\hat{V} = (\hat{v}_1, ..., \hat{v}_n)$ is a single-crossing order for E, then for each candidate $c \in C$ the set of voters who rank c first is a continuous subsequence of $\hat{V}$.*

Proof. Assume to the contrary that there exist two candidates $a, b \in C$ and integers k_1, k_2, and k_3, $1 \leq k_1 < k_2 < k_3 \leq n$, such that $\hat{v}_{k_1}$ and $\hat{v}_{k_3}$ rank a first while $\hat{v}_{k_2}$ ranks b first. Clearly, this means that $\hat{V}$ is not a single-crossing order for E (the relative order of a and b changes more than once as we consider the voters in the single-crossing order $\hat{V}$). □

The next theorem is the main technical tool for solving constructive candidate control problems under Plurality rule. In essence, it gives a polynomial-time algorithm for the problem of control by adding an exact number of candidates for Plurality, for the case, where adding a candidate cannot affect the score of our preferred candidate p.

Theorem 1 *For a given set of candidates C, a given set of unregistered candidates A, a given collection of voters V with preference over candidates from $C \cup A$, a given designated candidate $p \in C$ and a given nonnegative integer $k \leq \|A\|$, if $E = (C \cup A, V)$ is a single-crossing election and for each voter $v \in V$ who ranks p first among the candidates in C it holds that v ranks p first among the candidates in $C \cup A$, then the problem of determining whether there exists a set $A' \subseteq A$ of size k such that p is a unique winner of $E = (C \cup A', V)$ under* Plurality *rule is in* P.

Proof. We will give an algorithm based on dynamic programming.

Let $\hat{V} = (\hat{v}_1, \ldots, \hat{v}_n)$ be a single-crossing order of voters from V (for the candidate set $C \cup A$). For each candidate $c \in C \cup A$, let $first(c)$ and $last(c)$ be, respectively, the minimal and the maximal index of a voter from $\hat{V}$ that ranks c first in election $(C \cup \{c\}, V)$ (it is easy to see that without loss of generality, we can assume that these values are always well-defined[4]). From Lemma 1, we know that for a given $c \in C \cup A$ the interval $\hat{v}_{first(c)}, \ldots, \hat{v}_{last(c)}$ consist of all the voters from V which rank c first in $E = (C \cup \{c\}, V)$; let us refer to this interval later as the *visibility interval* of candidate c. For each set $X \subseteq C \cup A$ and each set $Y \subseteq V$, we define $maxscore(X, Y) = \max_{c \in X \setminus \{p\}} \left(score^P_{(X,Y)}(c)\right)$. We define a linear order L over the set of candidates A as follows: For each pair a, b of candidates from A, we have $a \; L \; b$ if and only if (by $\succ_{last(a)}$ we mean the preference order of voter $\hat{v}_{last(a)}$):

$$(last(a) < last(b)) \vee (last(a) = last(b) \wedge b \succ_{last(a)} a).$$

[4] Strictly speaking, such candidates might still be needed. Consider a case where A contains a single voter who is ranked last by all the voters, p is the current winner, and we ask if there is a set A' of size exactly one, such that p is a winner in $(C \cup A', V)$. We would have to take $A' = A$ and that would mean taking the candidate for whom *first* and *last* are not defined. However, with the way we use Theorem 1 in the paper, such a situation is never relevant. Handling this possibility is easy, but we omit it for the sake of clarity and brevity.

For a candidate $c \in A$, let $PreA(c)$ be the subset of A containing c and all the candidates that precede c under L. Similarly, for each voter $v \in V$, let $PreV(v)$ be the subset of V containing v and all candidates that are before v in single-crossing order $\hat{V}$. For each candidate $c \in A$ we write $PreV(c) = PreV(\hat{v}_{last(c)})$, thus $PreV(c)$ contains all the voters from $\hat{V}$ starting from the first one up to the last voter which ranks c first in election $E = (C \cup \{c\}, V)$.

For each integer u and each candidate $c \in A$, we define $f(u, c)$ as follows:

$$f(u,c) = \min_{\substack{A' \subseteq PreA(c) \\ \|A'\| = u}} \{maxscore(C \cup A', V)\}$$

Let a_{last} be the last candidate from A under L. Since no candidate from A can "steal" points from candidate p (this follows from our assumption on the input data), p can be made a winner of the election by adding exactly k candidates from A if and only if $f(k, a_{last}) < score^{P}_{(C,V)}(p)$.

We focus on computing $f(u, v)$ in polynomial-time. For each integer u, each candidate $c \in A$, and each voter $v \in V$, let $\mathcal{L}(u, c, v)$ be a family of cardinality-u subsets of $PreA(c)$ such that for each set X in $\mathcal{L}(u, c, v)$ it holds that: (a) $c \in X$, and (b) candidate c scores a point from voter v in election $(C \cup X, V)$. For a given integer u, a candidate $c \in A$, and a voter $v \in V$, we define a helper function g as follows:

$$g(u,c,v) = \min_{A' \in \mathcal{L}(u,c,v)} \{maxscore(C \cup A', PreV(v))\}$$

For a given candidate $c \in A$ and a given integer u, the expression for $g(u, c, \hat{v}_{last(c)})$ is almost identical to $f(u, c)$, except that it considers only subsets of $PreA(c)$ that contains c and $maxscore$ is computed on a limited set of voters. That is because voter $\hat{v}_{last(c)}$ ranks c first across all the candidates from $C \cup PreA(c)$, which follows from the definition of L. Therefore, to express f using g correctly, we need to scan through all possible candidates $c' \in PreA(c)$ so that we are not only limited to subsets of $PreA(c)$ containing c. We also need to keep in mind that function f considers all the voters from V in order to compute $maxscore$. Since for each $a \in A$ voters from $V \setminus PreV(a)$ do not rank candidates from $PreA(a)$ at the top, we can easily take care of the differences between g and f and express the latter using the former:

$$f(u,c) = \min_{c' \in PreA(c)} \{\max(g(u, c', \hat{v}_{last(c')}), maxscore(C, V \setminus PreV(c')))\} \quad (1)$$

This means that it suffices to be able to compute g in polynomial time to be able to also compute f in polynomial time.

We now describe a recursive method for computing g (we present this method as Algorithm 1, though the algorithm references the conditions we list below, so the reader might want to consider the algorithm and the description here in parallel). For each integer u, candidate $c \in A$, and voter $v \in V$, in order to compute $g(u, c, v)$ we scan through candidates $c' \in PreA(c)$, such that c scores a point from v in election $E = (C \cup \{c, c'\}, V)$, compute certain values (see below), and eventually return the smallest one of them. Let us fix $c \in A$ and $v \in V$. For each possible $c' \in PreA(c)$, such that c scores a point from v in $E = (C \cup \{c, c'\}, V)$ we consider the following cases:

Condition 1 ($last(c') < first(c)$). In this case the visibility intervals of c and c' dos not overlap and we return the maximum of $g(u-1, c', last(c'))$ and $maxscore(C \cup \{c\}, PreV(c) \setminus PreV(c'))$.

Condition 2 ($first(c') > first(c)$). In this case the visibility interval of c' is within the visibility interval. Since V is single-crossing, we know that if both c and c' take part in the election, candidate c' will score no points. We skip this case for now and take it into account later,

Condition 3 ($last(c') \geq first(c)$). In this case the visibility intervals of c and c' overlap. We return the maximum of $g(u - 1, c', t_{c',c})$ and $maxscore(C \cup \{c\}, PreV(c) \setminus PreV(t_{c',c}))$, where $t_{c',c}$ is such that all the voters $\hat{v}_1, \ldots, \hat{v}_{t_{c',c}}$ prefer c' to c, and the remaining ones prefer c to c'.

Going back to Condition 2, we know that for each c' that satisfies this condition, adding c' has no impact on the scores of the other candidates as long as c is taking part in the election (the visibility interval of c covers the visibility interval of c'). Thus, whether we add such a candidate to the election or not, has no impact on the value of g, except that we might want to add such candidates to ensure that we add exactly u candidates and not any less. Algorithm 1 shows how to recursively compute g referring to all the possible cases described above. All border cases are covered and we take Condition 2 into account in the following way: We compute the number t of candidates that satisfy it and allow the recursive calls to g to use between $u - 1$ and $u - 1 - t$ candidates (modeling the fact that we can add up to t candidates satisfying Condition 2). The algorithm returns ∞ if no subset $A' \subseteq A$ satisfying the definition of g exists.

Algorithm 1 Compute $g(u, c, v)$ recursively

```
if c does not score point from v in E = (C ∪ {c}, V) then
    return ∞
else if u = 1 then
    return maxscore(C ∪ {c}, PreV(v))
end if
result ← ∞
t ← the number of indices i' in {1, ..., i} that satisfy Condition 2
for c' ∈ PreA(c) \ {c} do
    if c scores a point from v in E = (C ∪ {c, c'}, V) then
        r' ← ∞
        if Condition 1 is satisfied then
            r' ← min_{t' ∈ {0,...,t}} {max(g(u − 1 − t', c', last(c'));
                    maxscore(C ∪ {c}, PreV(c) \ PreV(c'))}
        else if Condition 3 is satisfied then
            r' ← min_{t' ∈ {0,...,t}} {max(g(u − 1 − t', c', t_{c',c});
                    maxscore(C ∪ {c}, PreV(c) \ PreV(t_{c',c}))}
        end if
        result ← min(result; r')
    end if
end for
return result
```

Given this recursive formulation, we use standard *dynamic programming* techniques to compute g in polynomial time (strictly speaking, for each nonnegative integers u, i, s, where $u, i \leq \|A\|$ and $1 \leq s \leq n$, $g(u, i, s)$ is computable in time $O\left(n^2\|A\|^3\right)$). Using Equation (1), we compute $f(u, i)$ for each nonnegative u, i, where $u, i \leq \|A\|$, in time $O\left(n^2\|A\|^4\right)$. As per our discussion at the beginning of the proof, being able to compute f suffices to solve our problem. □

Based on the above theorem, it is relatively easy to prove our main result of this section (we omit the details due to space restriction).

Theorem 2 *Plurality-CCAC and Plurality-CCDC for single-crossing elections are both in* P.

3.2 Destructive Cases

Let us now consider the case of destructive candidate control under Plurality. In principle, one could derive polynomial-time algorithms for this case based on the constructive ones. However, dedicated algorithms can be much faster. The reason for this comes from the following result (we omit all proofs due to space constraints).

Theorem 3 *If destructive control by adding candidates is possible in single-crossing election under Plurality, then it can be achieved by adding at most three candidates from the set of unregistered candidates.*

In effect, to solve Plurality-DCAC it suffices to try all up-to-three-elements subsets of candidates to add. This leads to an algorithm that is much faster than the one for the constructive case.

Theorem 4 *Plurality-DCAC and Plurality-DCDC for single-crossing elections are both in* FP.

4 Condorcet Elections

For Condorcet and the unrestricted preference orders, candidate control and destructive voter control problems are easy, but constructive voter control problems are NP-complete [1, 18] (also see Table 1). We show that for the single-crossing case, the constructive voter control problems become easy as well (see Table 1). We note that the proof below is, in essence, a computational variant of the proof that the median voter's top-ranked candidate is always a Condorcet winner in single-crossing elections with an odd number of voters; see the paper of Rothstein [26] and the discussion in the work of Saporiti and Tohmé [27].

Theorem 5 *Condorcet-CCAV and Condorcet-CCDV for single-crossing elections are both in* P.

Proof. Due to space restriction, we consider the case of adding voters only. Before proving the actual theorem let us focus on a more general problem related to Condorcet-CCAV.

We are given a set of candidates C, a set of registered voters V of size n, a set of unregistered voters W, and a designated candidate $p \in C$. Let $U = V \cup W$ and let $s = \|U\|$. We are given a single-crossing order $\hat{U} = (\hat{v}_1, ..., \hat{v}_s)$ of the voters in U. We show a polynomial time algorithm for finding a set $W' \subseteq W$ of the smallest possible size, such that p is a *Condorcet winner* of election $(C, V \cup W')$.

For each pair of candidates $a, b \in C$, let $t_{a,b}$ be the integer such that $\{i \in [s] \mid a \succ_{\hat{v}_i} b\} = [t_{a,b}]$. For a set of voters $X \subseteq V \cup W$ and an integer $\ell \in [s]$, let $r(X, \ell)$ be the number of voters from X that are also in the sequence $(\hat{v}_1, ...\hat{v}_\ell)$. That is, $r(X, \ell) = \sum_{i \in [\ell]} \mathbf{1}_X(\hat{v}_i)$. Let $C_B = \{c \in C \mid c \succ_{\hat{v}_1} p\}$ and $C_W = \{c \in C \mid p \succ_{\hat{v}_1} c\}$. For each set $W' \subseteq W$ and each candidate $a \in C_W$, it is clear that in election $(C, V \cup W')$ candidate p is preferred over a by exactly $r(V \cup W', t_{p,a})$ voters. (It is easy to form an analogous condition for the candidates in C_B.) In order for p to be a *Condorcet winner* of $(C, V \cup W')$, p has to be preferred over any other candidate from C by more than half of the voters, which gives us a necessary and sufficient condition for p to win:

Condition 1. $r(V \cup W', t_{p,c}) > \frac{n+\|W'\|}{2}$ holds for each $c \in C_W$, and

Condition 2. $r(V \cup W', t_{c,p}) < \frac{n+\|W'\|}{2}$ holds for each $c \in C_B$.

Note that regardless of the choice of W', for each pair of integers $x, y \in [s]$ (where $x < y$), we have $r(V \cup W', x) \leq r(V \cup W', y)$. This means that it is sufficient to verify the two conditions above for the following pair of candidates only: For $c_W = \arg\min_{c \in C_W}(t_{p,c})$ and for $c_B = \arg\max_{c \in C_B}(t_{c,p})$. Let $t_W = t_{p,c_W}$ and $t_B = t_{c_B,p}$, we may now consider two scenarios. If $t_W < t_B$, then it is impossible to make p a *Condorcet winner* by adding voters. That is because for every $W' \subseteq W$ by combining the necessary conditions from above we get:

$$r(V \cup W', t_W) > \frac{n + \|W'\|}{2} \quad \text{and} \quad r(V \cup W', t_B) < \frac{n + \|W'\|}{2}.$$

This means that $r(V \cup W', t_W) > r(V \cup W', t_B)$, which contradicts with the fact that $t_W < t_B$. On the other hand, when $t_W > t_B$, Algorithm 2 will lead us to an optimal solution. If Condition 1 is not met for c_W in election E, the only option to satisfy it is by adding some number of voters from $(\hat{v}_1, ..., \hat{v}_{t_W})$ to the election. Symmetrically, when Condition 2 is not met for c_B, we should add voters from $(\hat{v}_{t_B+1}, ..., \hat{v}_s)$. In any case, it is always safe to add any number of voters from $(\hat{v}_{t_B+1}, ..., \hat{v}_{t_W})$ to the election with no bad influence on neither of the conditions (indeed, these voters rank p first). Clearly, if both conditions are not met, it is not possible to fulfill both of them by adding voters from $(\hat{v}_1, ..., \hat{v}_{t_B}, \hat{v}_{t_W+1}, ..., \hat{v}_s)$. Thus, after adding the voters from $(\hat{v}_{t_B+1}, ..., \hat{v}_{t_W})$, if both conditions are not met then it means that p cannot become a winner by adding voters from W. However, if only a single condition is not fulfilled, we can add voters from $(\hat{v}_{t_W+1}, ..., \hat{v}_s)$ or from $(\hat{v}_1, ..., \hat{v}_{t_B})$, respectively. Algorithm 2 is a greedy method for finding the subset of unregistered voters W' of the minimum size such that p is the Condorcet winner of $(C, V \cup W')$. In case where no solution exists, the algorithm returns **nil**. Clearly we can determine whether a solution exists and find it in polynomial time.

Algorithm 2 Condorcet-CCAV

```
if t_W < t_B then
    return nil
end if
W' ← ∅
while p is not a winner of election (C, V ∪ W') do
    v ← nil
    if exists voter from (v̂_{t_B+1}, ..., v̂_{t_W}) that is not in V ∪ W'
then
        v ← voter from (v̂_{t_B+1}, ..., v̂_{t_W}) that is not in V ∪ W'
    else if Condition 1 is not satisfied and Condition 2 is not
satisifed then
        return nil
    else if Condition 1 is not satisifed and exists voter from
(v̂_{t_W+1}, ..., v̂_s) that is not in V ∪ W' then
        v ← voter from (v̂_{t_W+1}, ..., v̂_s) that is not in V ∪ W'
    else if Condition 2 is not satisifed and exists voter from
(v̂_1, ..., v̂_{t_B}) that is not in V ∪ W' then
        v ← voter from (v̂_1, ..., v̂_{t_B}) that is not in V ∪ W'
    end if
    if v is not nil then
        W' ← W' ∪ {v}
    else
        return nil
    end if
end while
return W'
```

Let us now focus on Condorcet-CCAV for single-crossing election. In addition to the input data in the problem above, we are given an integer k—the size limit of $W' \subseteq W$. Let $I = (C, V, W, \hat{U}, p, k)$ be the instance of that problem. Using the method described above we can find $W' \subseteq W$ of the smallest possible size such that p is *Condorcet winner* in election $(C, V \cup W')$ in polynomial time (or conclude that no solution exists). If solution W' exists and $\|W'\| \leq k$ then we accept I, and otherwise we reject. Thus Condorcet-CCAV for the case of single-crossing elections is in P. □

5 Conclusions

We have studied the complexity of control for Plurality and Condorcet voting rules, for the case of elections that satisfy the single-crossing assumption. We have shown that, as in the case of single-peaked elections, all our control problems turn out to be polynomial-time solvable (whereas without assumptions on the nature of voters preferences candidate control problems are NP-complete for Plurality and constructive voter control problems are NP-complete for Condorcet's rule). We believe that our results are significant for at least two reasons. First, they provide another argument that NP-hardness results regarding election problems for the unrestricted domain are a weak sign of their practical difficulty. Second, currently there are only few computational results regarding the complexity of single-crossing elections and it is useful to provide results and techniques that can be useful for this domain restriction. The most natural open problem stemming from our work regards the complexity of control for Approval rule, for the case of a natural, to be discovered, notion of single-crossing approval elections. We are currently exploring this research direction.

ACKNOWLEDGEMENTS

We would like to thank the reviewers for a very useful set of comments. This work was supported in part by NCN grants 2012/06/M/ST1/00358 and 2011/03/B/ST6/01393, and by the Polish Ministry of Science and Higher Education (under AGH University Grant 11.11.230.015 (statutory project)).

REFERENCES

[1] J. Bartholdi, III, C. Tovey, and M. Trick, 'How hard is it to control an election?', *Mathematical and Computer Modeling*, **16**(8/9), 27–40, (1992).
[2] D. Black, *The Theory of Committees and Elections*, Cambridge University Press, 1958.
[3] F. Brandt, M. Brill, E. Hemaspaandra, and L. Hemaspaandra, 'Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates', in *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pp. 715–722. AAAI Press, (July 2010).
[4] R. Bredereck, J. Chen, and G. Woeginger, 'A characterization of the single-crossing domain', *Social Choice and Welfare*, **41**(4), 989–998, (October 2013).
[5] V. Conitzer, 'Eliciting single-peaked preferences using comparison queries', *Journal of Artificial Intelligence Research*, **35**, 161–191, (2009).
[6] D. Cornaz, L. Galand, and O. Spanjaard, 'Kemeny elections with bounded single-peaked or single-crossing width', in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 76–82, (2013).
[7] B. Dutta, M.O. Jackson, and M. Le Breton, 'Strategic candidacy and voting procedures', *Econometrica*, **69(4)**, 1013–1037, (2001).
[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, 'Rank aggregation methods for the web', in *Proceedings of the 10th International World Wide Web Conference*, pp. 613–622. ACM Press, (March 2001).
[9] E. Elkind, P. Faliszewski, and A. Slinko, 'Clone structures in voters' preferences', in *Proceedings of the 13th ACM Conference on Electronic Commerce*, pp. 496–513, (June 2012).
[10] E. Ephrati and J. Rosenschein, 'A heuristic technique for multi-agent planning', *Annals of Mathematics and Artificial Intelligence*, **20**(1–4), 13–67, (1997).
[11] G. Erdélyi and J. Rothe, 'Control complexity in fallback voting', in *Proceedings of Computing: the 16th Australasian Theory Symposium*, pp. 39–48. Australian Computer Society *Conferences in Research and Practice in Information Technology Series*, vol. 32, no. 8, (January 2010).
[12] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra, 'Using complexity to protect elections', *Communications of the ACM*, **53**(11), 74–82, (2010).
[13] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra, 'Multimode control attacks on elections', *Journal of Artificial Intelligence Research*, **40**, 305–351, (2011).
[14] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra, 'The complexity of manipulative attacks in nearly single-peaked electorates', *Artificial Intelligence*, **207**, 69–99, (February 2014).
[15] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe, 'Llull and Copeland voting computationally resist bribery and constructive control', *Journal of Artificial Intelligence Research*, **35**, 275–341, (2009).
[16] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe, 'The shield that never was: Societies with single-peaked preferences are more open to manipulation and control', *Information and Computation*, **209**(2), 89–107, (2011).
[17] S. Ghosh, M. Mundhe, K. Hernandez, and S. Sen, 'Voting for movies: The anatomy of recommender systems', in *Proceedings of the 3rd Annual Conference on Autonomous Agents*, pp. 434–435. ACM Press, (1999).
[18] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe, 'Anyone but him: The complexity of precluding an alternative', *Artificial Intelligence*, **171**(5–6), 255–285, (2007).
[19] H. Liu, H. Feng, D. Zhu, and J. Luan, 'Parameterized computational complexity of control problems in voting systems', *Theoretical Computer Science*, **410**(27–29), 2746–2753, (2009).
[20] T. Lu and C. Boutilier, 'Budgeted social choice: From consensus to personalized decision making', in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pp. 280–286, (2011).
[21] R. Meir, A. Procaccia, J. Rosenschein, and A. Zohar, 'The complexity of strategic behavior in multi-winner elections', *Journal of Artificial Intelligence Research*, **33**, 149–178, (2008).
[22] J. Mirrlees, 'An exploration in the theory of optimal income taxation', *Review of Economic Studies*, **38**, 175–208, (1971).
[23] C. Papadimitriou, *Computational Complexity*, Addison-Wesley, 1994.
[24] D. Parkes and L. Xia, 'A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs', in *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 1429–1435, (July 2012).
[25] T. Persson and G. Tabellini, *Political Economics: Explaining Economic Policy*, MIT Press, 2000.
[26] P. Rothstein, 'Representative voter theorems', *Public Choice*, **72**(2–3), 193–212, (December 1991).
[27] A. Saporiti and F. Tohmé, 'Single-crossing, strategic voting and the median choice rule', *Social Choice and Welfare*, **26**(2), 363–383, (2006).
[28] P. Skowron, L. Yu, P. Faliszewski, and E. Elkind, 'The complexity of fully proportional representation for single-crossing electorates', in *Proceedings of the 6th International Symposium on Algorithmic Game Theory*, pp. 1–12, (2013).
[29] T. Walsh, 'Uncertainty in preference elicitation and aggregation', in *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pp. 3–8. AAAI Press, (July 2007).