The Complexity of Priced Control in Elections

Tomasz Miąsko Piotr Faliszewski
AGH University of Science and Technology
Krakow, Poland

May 19, 2014

Abstract

We study the complexity of priced control in elections. Naturally, if a given control type is NP-hard for a given voting system $\mathcal{E}$ then its priced variant is NP-hard for this rule as well. It is, however, interesting what effect introducing prices has on the complexity of those control problems that without prices are tractable. We show that for four prominent voting rules (plurality, approval, Condorcet, and Copeland) introducing prices does not increase the complexity of control by adding/deleting candidates/voters. However, we do show an example of a scoring rule for which such effect takes place.

1 Introduction

We consider the complexity of election control [2, 25] for the case where different control actions can have possibly different prices. Our main motivation comes from the fact that different types of control actions allowed in multimode control reflect a wide range of ways in which elections can be influenced through political campaigns, and prices reflect the fact that the cost of different actions varies. Our main finding is that introducing prices in control problems, typically, does not change their complexity. Specifically, we show that for several well-known voting rules (plurality, approval, Condorcet, and Copeland) the complexity of control problems with prices remains the same as for the unpriced variants (however, showing this requires more care). On the other hand, we show that for scoring protocols destructive voter control is often easy, yet there is a scoring protocol for which destructive priced voter control is NP-hard. Our results stand in sharp contrast to those for control in weighted elections [16]. On one hand, allowing weighted votes often increases the complexity of control problems, and on the other, destructive weighted voter control for scoring protocols is always easy.

The individuals participating in an election, to whom we will refer as voters, might be, for example, members of parliaments, a jury, all adult citizens of a country, or even elements of distributed software systems [28, 20] or algorithms in various areas of computer science (we point to an application of voting related to natural language processing [27]). Voters select among possible alternatives, i.e., candidates taking part in the election. In the most frequently used, ordinal, model, a vote is a linear order over all the candidates, ranking them from the most to the least desirable. However, under approval voting voters simply indicate which candidates they do and do not approve of. Once all votes are gathered, we use a voting rule to determine the winner(s). There are many different voting rules to choose from, each with its own advantages and faults. For example, under plurality rule each candidate receives a point from each voter that ranks him or her first, and the candidate(s) with most points win. Under Copeland elections, for each two candidates we form a head-to-head contest (that is, we check which of the two is preferred by a majority of the voters), the winner receives a point, and whoever has most points in the end is the winner. We formally introduce all the voting systems studied in this paper in Section 2. We focus on four rules, plurality, approval, Condorcet, and Copeland, that are widely studied from the point of view of the complexity of control. This makes it easy to compare our results with other ones in the literature.

Since elections are used to decide on matters of great importance among individuals with conflicting preferences, it is no surprise that many agents are interested in influencing their outcomes. There are two basic goals that such agents may we willing to attain: either they try to ensure that a preferred candidate is the winner of the election (a constructive action) or they try to preclude a despised candidate from achieving a victory (a destructive action). Further, there are many ways in which voters, candidates, and election organizers can influence elections results. These ways range from strategic voting [22, 34] (see the survey of Faliszewski and Procaccia [19] for an AI-focused overview), through bribery [13], to running political campaigns [10, 9] and performing control attacks [2, 25]. We focus on the later two and we merge the ideas behind election control and campaign management.

By election control we mean actions that change the structure of an election. The most typical examples of control actions are adding/deleting candidates or voters. For example, it is easy to imagine settings where supporters of a particular candidate run a campaign to promote participating in an election, targeting the voters who are likely to vote for their candidate. Similarly, one can imagine actions discouraging opponent voters from casting their votes. A more difficult, yet possible, way of controlling an election is to fund a campaign of an additional candidate that would not otherwise take part in the election. Doing so could be motivated by a hope that such a candidate would steal votes away from our opponents.

The complexity of election control was first studied by Bartholdi, Tovey, and Trick [2] (we discuss related work in more detail in Section 1.1). However, they assumed that adding or deleting each candidate or voter has the same unit cost, which is not reasonable in the context of campaign management. Indeed, it might be very expensive to convince some candidate to join the race (e.g., because one would have to fund him or her completely), whereas convincing some other one might be quite cheap (because he or she already wants to join the election and is mostly prepared). Similarly, convincing some voters to vote may be more expensive than others (e.g., because for some we would have to pay for their transportation to the voting stations, whereas for others we would simply have to drop some leaflets in their neighborhood). Thus, in this paper we extend the model of control introduced by Bartholdi, Tovey, and Trick by allowing that different control actions have different prices. To simplify and shorten many of our proofs, we apply the multimode control framework of Faliszewski, Hemaspaandra, and Hemaspaandra [15].

1.1 Related Work

Computational study of election control was initiated by Bartholdi, Tovey, and Trick [2], who defined the problems of constructive election control by adding/deleting/partitioning candidates or voters for plurality and Condorcet elections. Later Hemaspaandra, Hemaspaandra, and Rothe [25] extended their work by considering destructive variants of these problems, and by also studying approval voting rule. Since these two papers, many researchers studied the complexity of control problems for various voting rules and in various other models. For example, Faliszewski et al. [17] considered the Copeland rule, Erdélyi et al. [12] studied Bucklin and fallback rules, and Parkes and Xia [32] studied Schulze's rule. This list, of course, is not exhaustive and is meant to present just a few examples (the reader may wish to consult survey of Faliszewski, Hemaspaandra and Hemaspaandra for some more details [14]).

In addition to studying election control for different voting rules, researchers extended the standard model of election control in many different directions. For example, Meir et al. [31] studied election control in multiwinner voting and introduced a model that generalizes the idea of constructive and destructive control. Faliszewski, Hemaspaandra, and Hemaspaandra [15] studied multimode control, where it is possible to perform several different types of control actions at the same time (e.g., it is possible to add some candidates, and delete some voters; the standard control problems allow one to either only add candidates or only delete voters, etc.). Faliszewski, Hemaspaandra, and Hemaspaandra [16] were the first to study control in weighted elections (however the work of Baumeister et al. [4] is related to this topic). Other authors took a different perspective and, for example, studied parametrized complexity of control problems [5, 29, 30, 36], or considered the complexity of control in elections where votes come from some restricted domains (focusing mostly on single-peaked elections [18, 7]). On the other hand, Wojtas and Faliszewski [35] studied counting variants of control problems, where instead of asking if someone can become a winner we ask for the probability that someone becomes a winner, given that a random control action is taken. This counting variant of control can be used to predict election winners and, thus, has similar applications as the research presented in this paper; it aims to guide the election campaigning process.

While traditionally election control problems are limited to adding/deleting/partitioning candidates and voters, there are many problems that are very close in spirit to election control. For example, Chevaleyre et al. [8] studied a differnt setting where new candidates can appear, and Elkind, Faliszewski, and Slinko [11] studied candidate cloning. Research on election control has also affected research on related fields. For example, Baumeister et al. [3] studied control in judgment aggregation.

Control Type	Adding Candidates		Deleting Candidates		Adding Voters		Deleting Voters	
Election System	**Const. Control**	**Dest. Control**	**Const. Control**	**Dest. Control**	**Const. Control**	**Dest. Control**	**Const. Control**	**Dest. Control**
Approval	I	**V**	**V**	**V**	R	**V**	R	**V**
Condorcet	I	**V**	**V**	**V**	R	**V**	R	**V**
Copeland	R	**V**	R	**V**	R	R	R	R
Plurality	R	R	R	R	**V**	**V**	**V**	**V**

Table 1: Summary of priced control vulnerabilities. Our results are typeset in boldface. In the table, we write I to indicate that the voting rule is *immune* to a given control attack, i.e., that it is impossible to affect the result by exerting this type of control. By R we mean that the rule is *resistant*, i.e., that it is not immune by the election control problem is NP-hard. By V we mean that the voting rule is *vulnerable*, i.e., that it is not immune and that there is a polynomial-time algorithm for the election control problem. Approval and Condorcet in our model are trivially vulnerable to destructive priced control by deleting candidates — the only possible successful control action is to delete despised candidate.

1.2 Results

Given the above, very rough, overview of literature on the complexity of election control, we see that there are three main lines of research regarding the topic. First, researchers seek complexity results for more and more different voting rules. Second, researchers seek to extend the election control model (e.g., by introducing weights, studying restricted domains, generalizing the notions of constructive/destructive control actions). Third, researchers apply the ideas from election control in other settings (e.g., in judgment aggregation).

Our paper follows the second line of research: We extend the model of election control by assuming that different control actions have possibly different costs. It is typical that papers that follow this second line of research focus on very few voting rules, usually including plurality, Condorcet, and approval. We follow this tradition as well. Specifically, we focus on these three voting rules and also consider Copeland. We show that for each of these rules the complexity of control by adding/deleting candidates or voters is the same irrespective if we assume that all control action have the same or possibly different costs. This result, however, is not trivial. Of course hardness proofs for the unit-cost model translate directly to hardness results in the model with varying costs, easiness results do not. Indeed, sometimes we have to replace very simple greedy algorithms with more involved ones, sometimes using dynamic programming. We summarize our results in Table 1.

Our results for the standard, prominent voting rules yield the question if adding prices can ever increase the complexity of control problems? We give an affirmative action by showing a scoring protocol for which destructive voter control is easy, but for which de-

[2] Approval and Condorcet in our model are trivially susceptible to destructive priced control by deleting candidates — the only possible successful control action is to delete despised candidate.

structive priced voter control is hard. This result answers our question, but leaves a bit more to be desired: Is there a natural voting rule witch such a property? We leave this as an open problem.

The current paper, in some sense, complements that of Faliszewski, Hemaspaandra, and Hemaspaandra [16] regarding the complexity of control in weighted elections. However, our results are very different. They show that adding weights to control problems often increases the complexity of control, whereas this is not the case for adding prices. They show that destructive weighted voter control under scoring protocols is always easy, whereas we show that destructive priced voter control can sometimes be NP-hard.

2 Preliminaries

In this section we review the ordinal model of elections, define voting rules that we study, and formally define (priced) election control problems.

Elections and Election Rules. An election $E = (C, V)$ consists of a set of candidates $C = \{c_1, \ldots, c_m\}$ and a set of voters $V = \{v_1, \ldots, v_n\}$. Voter's preferences are represented with strict total orders over the set of candidates. For example, if $C = \{a, b, c, d\}$ then some voter v_i might have preference order $d \succ c \succ b \succ a$, meaning that this voter likes d best, then c, then b, and finally he or she likes a least. For each election $E = (C, V)$ and each two candidates $c, d \in E$, we define $N_E(c, d)$ to be the number of voters in V who rank c above d.

An election rule $\mathcal{E}$ (voting rule, election system, voting system) is a function which given an election $E = (C, V)$ maps it to the set of winners $\mathcal{E}(E) \subseteq C$. In this paper we focus on polynomial-time computable voting rules but, in general, determining election winners can be a much more computationally demanding problem.[1] There can be more than one winner of an election. In such situations, to emphasize this fact, we refer to the winners of the election as the nonunique winners. Similarly, if there is only one winner of an election we refer to him or her as the unique winner. Finally, we allow situations where an election has no winners.

There are many different voting rules. For example, under plurality rule we give a point to each candidate that is ranked first, and choose as winners those candidates that have the highest number of points. More generally, a scoring rule is defined through a family of scoring vectors, one for each candidate-set cardinality, that define how many points a candidate receives for being ranked at a given position by a voter. Formally, a scoring vector (for an m-candidate election) is an m-tuple $\alpha = \langle \alpha_1, \ldots, \alpha_m \rangle$ of nonnegative integers given in nonincreasing order. For each vote where a candidate is ranked i'th, the candidate receives α_i points. Candidates that have the highest number of points are the winners. For example, plurality is defined through a family of scoring vectors of the form $\langle 1, 0, \ldots, 0 \rangle$.

[1] For example in a voting scheme suggested by Lewis Caroll checking if a distinguished candidate is a winner of an election is complete for parallel access to NP [1], [24]. The same holds for the systems of Young [33] and Kemeny [26]

Other interesting scoring protocols include, e.g., veto rule (defined through vectors of the form $\langle 1, \ldots, 1, 0\rangle$), k-approval rule (defined through vectors that start with k ones and then continue with zeros), k-veto (defined through vectors that end with k zeros, preceded by ones), and Borda rule (defined through vectors of the form $\langle m-1, m-2, \ldots, 0\rangle$).

In Condorcet method, a candidate c is a winner if he beats all the other candidates in head-to-head contests (i.e., if $N_E(c,d) > N_E(d,c)$ for all candidates d different from c). It is possible that there is no winner under Condorcet rule, but if there is one, he or she is unique.

Copeland's rule is an extension of Condorcet's rule in the sense that it elects Condorcet winner whenever it exists, and otherwise picks candidates that are close to being Condorcet winners in a certain way. Formally, for each rational α, $0 \leqslant \alpha \leqslant 1$, in the Copeland$^\alpha$ voting rule, candidate c receives one point for each candidate d, $d \neq c$, such that $N_E(c,d) > N_E(d,c)$, and α points for each candidate d, $d \neq c$, such that $N_E(c,d) = N_E(d,c)$. Candidates with the highest number of points are the winners. Naturally, there are many other rules that can be seen as extensions of Condorcet rule (e.g., maximin rule, Young rule, Kemeny rule, Dodgson rule; see, e.g., the overview of Brams and Fishburn [6]). However, among this type of rules, in this paper we focus on Copeland.

Finally, we also consider approval voting rule. Under approval, voters' preferences are represented differently. Instead of ranking the candidates, each voter provides a set of candidates that he or she approves of. A candidate receives a point for each voter that approves of him or her. As before, the candidates with the highest number of points are the winners.

We denote the score of a candidate c in election E by $\mathrm{score}_E(c)$ (the actual voting rule will always be clear from context). When the election E is clear from context, we sometimes write $\mathrm{score}(c)$ instead of $\mathrm{score}_E(c)$. Further, for an election $E = (C, V)$ and candidates $c, d \in C$, we write $\mathrm{diff}_E(c,d)$ to mean difference between the score of candidate c and the score of candidate d. For the case of Condorcet rule, by $\mathrm{diff}_E(c,d)$ we mean $N_E(c,d) - N_E(d,c)$.

Election Control Problems. We consider priced multimode control problems. In control problems an attacker tries to execute a basic control action such as candidate addition, candidate deletion, voter addition or voter deletion to change the result of an election. In priced multimode control problems several different types of basic control actions can be combined into a single attack. Moreover, each such action has associated price and person exercising control over the election has only a limited budget.

We assume that there is a price tag for each voter and candidate that we add or delete. That is, for each voter v that can be added or deleted, we have number $\Pi(v)$, the price of adding/deleting v. For each candidate c that can be added or deleted, we have number $\Pi(c)$, the price of adding/deleting c. To simplify notation, if we consider, e.g., some set W of voters, we write $\Pi(W)$ to mean $\sum_{v \in W} \Pi(v)$. We use analogous notation for candidate sets. Unless stated otherwise, we assume that the prices are encoded in binary.

With this notation available, we define the most general form of our control problem.

Name: $\mathcal{E}$-AC-DC-AV-DV-priced-control.

Given: An election (C, V), a candidate $c \in C$, a set of additional candidates D, such that $C \cap D = \emptyset$, a set of additional voters W, prices Π for candidates in $C \cup D$ and voters in $V \cup W$, and a natural number K (the budget).

Question (constructive): Are there subsets $C' \subseteq C$, $D' \subseteq D$, $V' \subseteq V$, $W' \subseteq W$, such that candidate c is the unique winner of $\mathcal{E}$ election $((C \setminus C') \cup D', (V \setminus V') \cup W')$ and $\Pi(C' \cup D') + \Pi(V' \cup W') \leqslant K$.

Question (destructive): Are there subsets $C' \subseteq C$, $D' \subseteq D$, $V' \subseteq V$, $W' \subseteq W$, such that candidate c is not the unique winner of $\mathcal{E}$ election $((C \setminus C') \cup D', (V \setminus V') \cup W')$ and $\Pi(C' \cup D') + \Pi(V' \cup W') \leqslant K$.

This definition calls for some comments. We note that subset C' is the set of candidates to be removed from the election and subset D' is the set of candidates to be added to the election. Analogously, V' is the set of deleted voters and W' is the set of added voters. The total price of such a control action is the sum of the prices of added/deleted) candidates and voters. That is, the total price is $\Pi(C' \cup D') + \Pi(V' \cup W')$. For the case where all the prices are equal to one, we refer to the above problem as $\mathcal{E}$-AC-DC-AV-DV-control (omitting the word "priced").

We point out that even though we follow the idea of multimode control of Faliszewski, Hemaspaandra, and Hemaspaandra [15], we slightly differ from their approach. Indeed, they have a separate "budget" for each control type, whereas we have a single parameter K that models the total budget. This matches our motivating example of campaign management better. If one is running a campaign, there is a single budget that can be partitioned between various activities in any convenient way.

We use the unique-winner model. That is, to be successful, a candidate has to be the only winner of the election. Both unique-winner model and nonunique-winner model are frequently studied in election control literature. While occasionally the choice of the particular model matters, in our work it is immaterial.

In the constructive cases, we will often speak of the distinguished candidate c as the *preferred* candidate and thus we will often denote him with p rather than with c. For the destructive cases, we will refer to this candidate as the *despised* one and often use d to denote him or her. In destructive control by deleting candidates it is usually assumed that the despised candidate cannot be removed from the election. This condition is necessary for the case where deleting each candidate has unit cost, because otherwise one could simply remove the despised candidate. On the other hand, in our model, with prices, we do not pose this requirement. If one does not want to allow the despised candidate to be deleted, one can set his or her deletion price to be above the available budget K.

We are often interested in subproblems of $\mathcal{E}$-AC-DC-AV-DV priced control where only some nonempty subset of basic control actions, AC (adding candidates), DC (deleting candidates), AV (adding voters) and DV (deleting voters), is available. We denote such subproblems by leaving only relevant parts of the input and appropriately modifying the question part of the problem. (Intuitively, one could also think that all the disallowed control actions have prices higher than the available budget.) Names of such control problems are formed from the name of the voting system $\mathcal{E}$, followed by the permitted basic control actions, where all parts are separated with "-" character. For example, if we studied priced control by adding candidates and deleting voters, then the control problem's name would be $\mathcal{E}$-AC-DV-priced-control.

We say that a voting system is *susceptible* to constructive control problem $\mathcal{C}$ if for some instance of this control problem with an election E, a preferred candidate is not a unique winner of the election E , but it is possible to exercise control $\mathcal{C}$ over the election E in such way as to make the preferred candidate the unique winner. Similarly, in the case of destructive control, voting system is susceptible to control if we can prevent the despised candidate from being the unique winner, and he or she had not been the unique winner before. A voting system is said to be *immune* to control if it is not susceptible to it. If a voting system is susceptible to control and associated decision problem is in P, then we say that the voting system is *vulnerable* to this type of control. If a voting system is susceptible to control and associated decision problem is NP-hard, we say it is *resistant* to this type of control.

The main goal of this paper is to establish the complexity of priced control by adding or deleting candidates or voters for plurality rule, approval rule, Condorcet rule, and Copeland rule. We focus on problems where only a single type of control is allowed, but occasionally we will use the expressive power of multimode control to simplify and compress our proofs.

Computational Complexity. We assume that the reader is familiar with basic notions of complexity theory such as classes P and NP, many-one reductions, and the notion of NP-completeness. However, most of the proofs in this paper present polynomial-time algorithms.

3 Prices Often Do Not Affect the Complexity of Control

In this section we study priced control problems under plurality, approval, Condorcet and Copeland rules, using the multimode control framework. Naturally, introducing prices cannot make our control problems easier and, indeed, the following easy proposition holds.

Proposition 3.1. For each voting rule $\mathcal{E}$ and each control type $\mathcal{C}$, it holds that constructive (destructive) $\mathcal{E}$-$\mathcal{C}$-control many-one reduces to constructive (destructive) $\mathcal{E}$-$\mathcal{C}$-priced-control.

Thus all the hardness results for the unpriced control problems hold in the priced setting. The main message of this section is that, nonetheless, all the existing vulnerability results for adding/deleting candidates or voters for our four voting rules do carry through to the setting with prices. In particular, we show that plurality is vulnerable to constructive and destructive AV-DV-priced-control, approval and Condorcet are vulnerable to destructive AC-DC-AV-DV-priced-control and constructive DC-priced-control, and Copeland is vulnerable to destructive AC-DC-priced-control. This suggests that adding prices would never increase the complexity of control problems. However, in Section 4 we show that in fact there are scoring protocols for which considering prices makes a difference in terms of the complexity of control problems.

In the following sections we present our results for plurality, approval, Condorcet, and Copeland rules. We remark that we will phrase our results in full generality, using the multimode control framework. However, of course, our results that regard several control types at the same time carry through to settings with fewer control types. For example, the fact that Condorcet rule is vulnerable to destructive AC-DC-AV-DV-priced-control means that it is also vulnerable to destructive variants of each of AC-priced-control, DC-priced-control, AV-priced-control, and DV-priced-control (this follows naturally from the definition because we can set up the prices for disallowed control actions to be above the allowed budget; a very similar result is given as Proposition 4.9 by Faliszewski, Hemaspaandra, and Hemaspaandra [15]).

3.1 Plurality Rule

We start by considering the plurality rule. Bartholdi, Tovey, and Trick [2] have shown that plurality is resistant to constructive control by adding/deleting candidates, but that it is vulnerable to constructive control by adding/deleting voters. Hemaspaandra, Hemaspaandra, and Rothe [25] have shown that the same results hold for the destructive cases of these problems. We extend the vulnerability results to the priced case by showing that Plurality-AV-DV-priced-control is in P both in the constructive and in the destructive case.

Plurality is a very simple rule. If a new vote is added to the election, then the score of candidate who is ranked first in this vote is increased by one, while the scores of all the other candidates remain intact. Similarly, when deleting a single vote from the election only the score of one candidate is affected. This locality property makes it possible to construct greedy algorithms for Plurality-AV-DV-priced-control. Our algorithms are natural extensions of those for the unpriced setting.

Theorem 3.2. *Constructive Plurality-*AV-DV*-priced-control is in* P*.*

Proof. Input to the constructive Plurality-AV-DV priced control problem consists of an election $E = (C, V)$, a preferred candidate $p \in C$, a set of additional voters W, a list of prices Π associated with the votes from $V \cup W$, and a natural number K (the available budget). We give a greedy algorithm which in each step decreases the value of $\max_{c \in C \setminus \{p\}} \text{diff}(c, p)$ by one, and halts either when we make p the unique winner of the election, or the available budget is exceeded, or there are no more votes to add/remove.

Our algorithm proceeds as follows. If p is already the unique winner, then accept. Otherwise keep executing one of the following actions, until the stop condition is met. There are two possible actions, the one with lower cost is selected and executed:

1. From the set of additional votes that rank p first, pick a vote w which has not been added to the election yet and which has minimal price $\Pi(w)$. The cost of this action is $\Pi(w)$. If this action is executed, add w to the election.

2. For each candidate c in $\arg\max_{c \in C} \text{diff}(c, p)$, pick a vote $v \in V$ that ranks c first, that has not already been deleted from the election, and that has minimal price among

Plurality-AV-DV-Constructive-Control(C, V, p, W, Π, K)

```
E ← (C, V)
V' ← W' ← ∅
while p is not the unique winner of E and (V \ V') ∪ (W \ W') ≠ ∅ do
    w ← a vote for p with minimal price from W \ W'
    U ← votes for candidates in arg max_{c∈C} diff(c, p) with minimal price, one for each
        candidate
    if Π(w) ⩽ Π(U) then
        W' ← W' ∪ {w}
        K ← K − Π(w)
    else
        V' ← V' ∪ U
        K ← K − Π(U)
    E ← (C, (V \ V') ∪ W')
    if K < 0 then reject
if p is the unique winner of E then accept else reject
```

Figure 1: The algorithm for constructive Plurality-AV-DV priced control problem.

such votes. Let U be the collection of the picked votes. The cost of this action is $\Pi(U)$. If this action is executed, all the votes from U are deleted from the election.

When the stop condition is met, we verify that p is the unique winner of the election and the budget is not exceeded. We accept or reject accordingly. The algorithm's pseudocode is presented on Figure 1. The correctness and polynomial running time are straightforward to see. $\square$

In constructive Plurality-AV-DV priced control we have to ensure that preferred candidate's score is higher than score of all remaining candidates. On the other hand, in destructive control we only have to ensure that there exists at least one candidate with score equal to or higher than the score of the despised candidate. This suggests a simple algorithm enumerating all candidates and checking if one of them can beat or tie the despised one. As before, this is a natural extension of algorithms for the unpriced setting.

Theorem 3.3. *Destructive Plurality-*AV-DV *priced control is in* P.

Proof. Input to the destructive Plurality-AV-DV priced control instance consists of an election $E = (C, V)$, the despised candidate $d \in C$, a set of additional voters W, a list of prices associated with voters $V \cup W$, and an available budget $K \in \mathbb{N}$. For each candidate $c \in C \setminus \{d\}$ we create a list of votes from V where d is ranked first, which we could delete from the election, and a list of votes in W where c is ranked first, which we could add to the election. Merge these lists together, sort in the order of increasing prices, and take first $\max(0, \mathrm{diff}(d, c))$ votes, i.e., the number of votes that creates a tie between candidate

```
Plurality-Destructive-Control(C, V, d, W, Π, K)
for c ∈ C \ {d} do
    V' ← list of votes for d in V
    W' ← list of votes for c in W
    L ← list V' ∪ W' sorted in the order of increasing prices
    L ← first max(0, diff(d, c)) votes from L
    if diff(d, c) ⩽ |L| and Π(L) ⩽ K then
        accept
reject
```

Figure 2: The algorithm for destructive Plurality-AV-DV priced control problem.

d and candidate c. If there is sufficiently many votes and their total price does not exceed available budget K, then accept. Otherwise if this condition is not fulfilled for any candidate $c \in C \setminus \{d\}$, reject. Pseudocode for this algorithm is presented on Figure 2. □

Thus, for the case of plurality, introducing prices is seamless; we can adjust the existing greedy algorithms in a simple way. We believe that this is a very positive results. Priced control problems are more realistic and it is convenient that considering prices comes at essentially no additional cost in terms of computational complexity.

3.2 Approval and Condorcet Rules

Let us now move on to the case of approval and Condorcet rules. We extend the results of Faliszewski, Hemaspaandra and Hemaspaandra [15] (who themselves relied on the results of Bartholdi, Tovey, and Trick [2] and Hemaspaandra, Hemaspaandra, and Rothe [25]), who have shown that approval and Condorcet rules are vulnerable to destructive AC-AV-DV-control, to apply to priced control. In fact, we show that approval and Condorcet are vulnerable to destructive AC-DC-AV-DV-priced-control because, as we have argued, in our model it makes sense to consider destructive control by deleting candidates where we can delete the despised candidate. (In the unpriced model such action is prohibited as it makes the problem trivial.) We also show that approval and Condorcet are vulnerable to constructive DC-priced-control. Naturally, through proposition 3.1 and the results of Bartholdi, Tovey, and Trick [2] and Hemaspaandra, Hemaspaandra, and Rothe [25], approval and Condorcet are resistant to all the remaining types of control.

The reader may wonder why we consider approval and Condorcet rules jointly. The reason is that both approval elections and Condorcet elections can be understood in terms of the results of head-to-head contests between candidates. By head-to-head contests we mean elections were only two candidates are present. To facilitate this approach we adopt the following convention: we say that candidate c is preferred to candidate d in election with voter set V if and only if $\mathrm{diff}_{(\{c,d\},V)}(c,d) > 0$ (recall that for approval rule $\mathrm{diff}_E(c,d)$

means the difference of scores of candidates c and d in election E, whereas for Condorcet it means the value $N_E(c,d) - N_E(d,c))$.

With this notation available, we see that a candidate is the unique winner of an approval election or of a Condorcet election if and only if he or she is the unique winner of all the head-to-head contests with the other candidates. Thus to prevent a despised candidate d from being a unique winner we can either:

1. delete the despised candidate from the election, or

2. ensure that another candidate beats or ties the despised candidate in their head-to-head contest.

In the second case, the despised candidate's loss to some candidate c can be achieved by introducing voters who prefer c to d, or by deleting voters who prefer d to c. Each such added or deleted voter introduces the same change in the score difference (difference in number of approvals) between candidate c and the despised candidate. This observation suggest a simple algorithm based on enumeration of candidates who might ensure despised candidate's defeat, combined with a greedy approach to selecting the votes relevant to the head-to-head contest with the despised candidate.

Theorem 3.4. *Approval voting and Condorcet voting are vulnerable to destructive* AC-DC-AV-DV *priced control.*

Proof. In destructive AC-DC-AV-DV priced control instance we are given an election $E = (C, V)$, a despised candidate $d \in C$, a set of additional candidates D, a set of additional voters W, prices Π associated with candidates $C \cup D$ and voters $V \cup W$, and an available budget $K \in \mathbb{N}$.

If candidate d already is not a unique winner of election E or we can remove d from election (i.e., $\Pi(d) \leqslant K$) then control is successful and we accept. Otherwise, for each candidate $c \in (C \cup D) \setminus \{d\}$, we create a list L containing votes from V, where d is preferred to c, and votes form W, where c is preferred to d. We sort L in the order of increasing prices, and limit it to the first $\mathrm{diff}_E(d,c)$ votes. In our control action we delete the votes from V that are in L, and add the votes from W that are in L. Therefore the total price of control action is the sum over:

1. Prices associated with votes added and removed to the election.

2. The price of adding c to the election, if $c \in D$.

If there are enough votes to create a tie between candidate c and candidate d, and total cost does not exceed K, accept. Otherwise, repeat this procedure for all remaining candidates. If control is not possible for any candidate $c \in (C \cup D) \setminus \{d\}$ reject. The final algorithm is presented on Figure 3. □

In the constructive setting, approval and Condorcet rules are vulnerable to control by deleting candidates only. (They are immune to control by adding candidates and resistant

```
Approval-Condorcet-Destructive-AC-DC-AV-DV-Control(C, V, d, D, W, Π, K)
E ← (C, V)
if d is not the unique winner of E or Π(d) ⩽ K then
    accept
for c ∈ (C ∪ D) \ {d} do
    V′ ← list of votes in V, where d is preferred to c
    W′ ← list of votes in W, where c is preferred to d
    L ← list V′ ∪ W′ sorted in the order of increasing prices
    L ← first diff_E(d, c) votes from L
    K′ ← Π(L)
    if c ∈ D then
        K′ ← K′ + Π(c)
    if diff_E(d, c) ⩽ |L| and K′ ⩽ K then
        accept
reject
```

Figure 3: The algorithm for destructive AC-DC-AV-DV priced control in Approval voting and Condorcet voting.

to voter control [2, 25].) We extend this result to the priced setting. Clearly, to make the preferred candidate win, all the candidates that defeat him or her in their head-to-head contest should be deleted. Furthermore, as deleting candidates does not affect in any way the results of head-to-head contests, it is a necessary and sufficient condition. There is single optimal control action.

Theorem 3.5. *Approval voting and Condorcet voting are vulnerable to constructive* DC *priced control.*

Proof. In constructive DC priced control instance we are given an election $E = (C, V)$, a preferred candidate $p \in C$, a list of prices associated with each candidate in C, and an available budget K. Candidate p is the unique winner of the election E if and only if he or she beats all remaining candidates in their head-to-head contests. Therefore, if total price necessary to delete all candidates who tie or beat the preferred candidate p in a head-to-head contests is within budget, then accept, otherwise reject. Pseudocode is presented on Figure 4, □

Again, introducing prices does not make the control problems significantly harder for approval and Condorcet rules. It is easy and natural to extend existing greedy algorithms to take prices into account. As we will see in the next section, the case of Copeland is somewhat more involved.

Approval-Condorcet-Constructive-DC-Control(C, V, p, Π, K)

$E \leftarrow (C, V)$
$C' \leftarrow \{c \in C \setminus \{p\} \mid \mathrm{diff}_E(p, c) \leqslant 0\}$
if $\Pi(C') \leqslant K$ **then**
    accept
else
    reject

Figure 4: The algorithm for constructive DC priced control in Approval voting and Condorcet voting.

3.3 Copeland Rule

Faliszewski at al. [17] have studied the Llull and Copeland voting and have shown that Copeland$^\alpha$ voting fully resists constructive control and, among basic types of control (AC, DC, AV and DV), is vulnerable to destructive AC and destructive DC control only. These vulnerability results have been combined into destructive AC-DC control vulnerability by Faliszewski, Hemaspaandra, and Hemaspaandra [15]. Here we extend this result to the priced control framework.

In the following theorem we extend the algorithm of Faliszewski, Hemaspaandra, and Hemaspaandra [15, Theorem 4.10] to the case of destructive AC-DC-priced-control, for the case of Copeland^0 and Copeland^1 rules. Then we explain why it does not work for Copeland$^\alpha$ for all rational α values, α, $0 \leqslant \alpha \leqslant 1$. Finally, in Theorem 3.8, we provide an algorithm which does work for all rational values of α.

Theorem 3.6. *Destructive* AC-DC*-priced-control is in* P *for Copeland*^{0} *and Copeland*^{1} *voting.*

Proof. In destructive Copeland$^\alpha$-AC-DC priced control instance, where α is in $\{0, 1\}$, we are given an election $E = (C, V)$, a despised candidate $d \in C$, a set of additional candidates D, a list of prices Π associated with the candidates in $C \cup D$, and the available budget $K \in \mathbb{N}$. To preclude despised candidate d from being the unique winner of the election, apart from obvious action of prohibiting him or her from taking part in the election, we can ensure that the score of another candidate from $C \cup D$, call him or her p (we try each possible choice of p), is higher or equal to the score of d, i.e., $\mathrm{diff}(d, p) \leqslant 0$. The score of a candidate in Copeland$^\alpha$ election is a sum of his or her scores in head-to-head contents with the remaining candidates:

$$\mathrm{score}_{(C,V)}(d) = \sum_{c \in C \setminus \{d\}} \mathrm{score}_{(\{c,d\},V)}(c)$$

For each candidate c, define $\mathrm{gain}(c)$ to be the score difference that candidate p gains relative

```
Copeland01-Destructive-AC-DC-Control(C, V, d, D, Π, K)
E ← (C, V)
if d is not the unique winner of E or Π(d) ⩽ K then
    accept
for p ∈ (C ∪ D) \ {d} do
    C′ ← {c ∈ C \ {p, d} | gain(c) > 0}
    D′ ← {c ∈ D \ {p} | gain(c) > 0}
    L ← list C′ ∪ D′ sorted in the order of increasing prices
    L ← first max(0, diff_E(d, p)) votes from L
    K′ ← Π(L)
    if p ∈ D then
        K′ ← K′ + Π(p)
    if diff_E(d, p) ⩽ |L| and K′ ⩽ K then
        accept
reject
```

Figure 5: The algorithm for destructive AC-DC priced control in Copeland^0 and Copeland^1 voting.

to the despised candidate d, if candidate c is part of control action:

$$\text{gain}(c) = \begin{cases} \text{score}_{(\{c,d\},V)}(d) - \text{score}_{(\{c,p\},V)}(p), & c \in C \\ \text{score}_{(\{c,p\},V)}(p) - \text{score}_{(\{c,d\},V)}(d), & c \in D \end{cases}$$

It easy to see that for Copeland^0 and Copeland^1, for each candidate c, $\text{gain}(c)$ is either $-1, 0$ or 1. Moreover, as our goal is to decrease d's advantage over p, we are only interested in candidates with positive gain. Consequently, the following greedy approach can be used to select candidates to add or delete. From $C \setminus \{d, p\}$ and $D \setminus \{p\}$ select a list L of candidates with positive gain. Sort L in the order of increasing prices. Take first $\max(0, \text{diff}_E(d, p))$ candidates from L and if there was a sufficient number of them, then $A = L \cup (D \cap \{p\})$ describes a successful control action. If the total price of control action A is within budget K then accept, otherwise repeat this procedure for another choice of candidate p. The final algorithm is presented on Figure 5. □

The above algorithm relies on the fact that all the candidates that we add or remove from the election introduce the same score difference between the despised candidate and our chosen candidate p. This is a crucial element ensuring correctness of the greedy approach. In Copeland^α for some rational α, $0 < \alpha < 1$, the score difference could be 1, α or $1 - \alpha$. This makes the candidates incomparable and the greedy approach infeasible.

Now we reformulate Copeland^α into voting system $\text{Copeland}^{x,y}_{\mathbb{N}}$ that admits only natural numbered scores to facilitate our dynamic programming solution.

Definition 3.7. For each $x, y \in \mathbb{N}$, and an election $E = (C, V)$. Candidate with the highest score is a winner of the $\text{Copeland}_{\mathbb{N}}^{x,y}$ election, where the score of candidate c is defined to be:

$$\begin{aligned}\text{score}_E(c) = x\,&|\{d \in C \setminus \{c\} \mid N_E(c,d) > N_E(d,c)\}| \\ +y\,&|\{d \in C \setminus \{c\} \mid N_E(c,d) = N_E(d,c)\}|\end{aligned}$$

It is easy to see that for each rational α, $0 < \alpha \leqslant 1$, Copeland^α election is equivalent to the $\text{Copeland}_{\mathbb{N}}^{x,y}$ election where $\alpha = y/x$, for some $x, y \in \mathbb{N}$.

Theorem 3.8. *For each $x, y \in \mathbb{N}$, destructive $\text{Copeland}_{\mathbb{N}}^{x,y}$-AC-DC-priced-control is in* P.

Proof. The input to the destructive $\text{Copeland}_{\mathbb{N}}^{x,y}$-AC-DC-priced-control problem consists of an election $E = (C, V)$, a set of additional candidates D, a despised candidate $d \in C$, a list Π of prices associated with candidates $C \cup D$ and available budget $K \in \mathbb{N}$. If d is already not a unique winner of the election or $\Pi(d) \leqslant K$ then accept. Otherwise for each candidate $p \in C \cup D$ distinct from d, we check if it is possible to ensure that $\text{diff}(p, d) \geqslant 0$ by executing some control action within budget. Let $A = (C \cup D) \setminus \{d, p\} = \{a_1, \ldots, a_n\}$ and define $m(i, g)$ to be the minimal price of control necessary to achieve a total gain (as defined in the previous proof) of at least g by executing control actions involving at most candidates from the set $\{a_1, \ldots, a_i\}$. It is easy to see that following recursive relations hold:

$$m(i,g) = \begin{cases} 0 & \text{if } i = 0 \text{ and } g = 0 \\ \infty & \text{if } i = 0 \text{ and } g \neq 0 \\ \min\left(m(i-1,g), \Pi(a_i)\right) & \text{if } i > 0 \text{ and } g \leqslant \text{gain}(a_i) \\ \min\left(m(i-1,g), m(i-1, g - \text{gain}(a_i)) + \Pi(a_i)\right) & \text{if } i > 0 \text{ and } g > \text{gain}(a_i) \end{cases}$$

Candidate p can beat or tie the despised candidate d if and only if $m(n, \text{diff}_E(d,p)) + \chi_D(p) \cdot \Pi(p) \leqslant K$ (where $\chi_D(p)$ is 1 if $p \in D$ and is 0 otherwise). We can compute the value of $m(n, \text{diff}_E(d,p))$ in polynomial time using standard dynamic programming techniques. □

Corollary 3.9. For each rational α, $0 \leqslant \alpha \leqslant 1$, destructive Copeland^α-AC-DC priced control is in P.

Proof. For each rational α, $0 < \alpha \leqslant 1$, this follows directly from Theorem 3.8. The missing case of $\alpha = 0$ is provided by Theorem 3.6. □

The above discussion shows that algorithms for priced control are not always simple extensions of those for the unpriced cases, and indeed can require new ideas. In the next section we show that it is even possible that introducing prices moves control problems from being solvable in polynomial time to being NP-hard.

4 Prices Can Increase the Complexity of Control

In the previous section, we have shown that for several prominent voting rules introducing prices does not affect the complexity of control problem. Now we will show that, nonetheless, there are rules for which it is not the case. First, we show that destructive AV-priced-control and DV-priced-control problems are polynomial-time solvable for scoring rules, provided that either the prices or the entries of used scoring vectors are encoded in unary. In particular, it means that for every scoring protocol, destructive control by adding/deleting voters (without prices) is in P. Second, we show an example of a scoring protocol, whose entries are encoded in binary, for which destructive Scoring-AV-priced-control and Scoring-DV-priced-control problems are NP-hard. It is particularly interesting that our proof follows by a reduction from the X3C problem rather than from the Partition problem, as is often the case for election problems with binary-encoded prices/weights.

It is interesting to compare our results to those of Faliszewski, Hemaspaandra, and Hemaspaandra [16] regarding control problems in weighted elections. While they mostly consider constructive cases, they remark that destructive voter control for scoring protocols in weighted elections is in P (in weighted elections for each voter v there is a natural number w_v, his or her weight, and we treat the vote of v as if it was cast by w_v voters with the same preference order). Our results show that the complexity of destructive voter control for the case of priced elections behaves in much more intricate ways.

4.1 Vulnerability Results

We first provide our vulnerability results. To simplify the proofs, we define the following head-to-head priced control problem in which we ask if some specific candidate can tie or beat the despised candidate by adding voters to the election.

Name: Scoring head-to-head priced control.

Given: An election (C, V), a scoring vector $\alpha = \langle \alpha_1, \ldots, \alpha_{|C|} \rangle$, a despised candidate $d \in C$, a preferred candidate $p \in C$ distinct from d, a set of additional voters W, a list of natural numbers Π describing prices associated with voters W, and available budget $K \in \mathbb{N}$.

Question: Is there a subset $W' \subseteq W$ such that $\mathrm{diff}_{(C,V \cup W')}(p, d) \geqslant 0$ and $\Pi(W') \leqslant K$.

In scoring head-to-head priced control, candidates' scores in the election are calculated using given scoring vector α.

Lemma 4.1. *Scoring head-to-head priced control is in* P *if scoring vectors entries are represented in unary.*

Proof. In scoring head-to-head priced control by voters addition we are given an election $E = (C, V)$, a scoring vector α, distinguished candidate d, a preferred candidate $p \in C \setminus \{d\}$, a set of additional voters W with their prices Π and available budget $K \in \mathbb{N}$. We assume that $\mathrm{diff}_E(d,p) > 0$; otherwise the problem is trivial. Define $\mathrm{gain}_C(v)$ to be $\mathrm{diff}_{(C,\{v\})}(p,d)$, i.e., the score difference that p gains relative to d if we add voter v to the election E. We observe that it is of no use to add voters with nonpositive gain if we try to increase score difference between p and d. Let $W' = \{w_1, \dots, w_{|W'|}\}$ be a set of voters from W with positive gain. Let $m(i,g)$ be the minimal price of control necessary to achieve a total gain (summed over all votes we decided to add) equal to or higher than g, using first i voters from W'. If it is not possible to achieve such gain, define $m(i,g)$ to be infinite. The value $m(i,g)$ can be computed using the following recursive definition:

$$m(i,g) = \begin{cases} 0 & \text{if } i = 0 \text{ and } g = 0 \\ \infty & \text{if } i = 0 \text{ and } g > 0 \\ \min\left[m(i-1,g),\ \Pi(v_i)\right] & \text{if } i \geqslant 1 \text{ and } g \leqslant \mathrm{gain}_C(v_i) \\ \min\left[m(i-1,g),\ m(i-1, g - \mathrm{gain}_C(v_i)) + \Pi(v_i)\right] & \text{if } i \geqslant 1 \text{ and } g > \mathrm{gain}_C(v_i) \end{cases}$$

Candidate p can tie or beat d if and only if $m(|W'|, \mathrm{diff}_E(d,p)) \leqslant K$, which can be checked in polynomial time using standard dynamic programming. Polynomial running time follows from the fact that the entries in vector α are represented in unary. □

Lemma 4.2. *Scoring head-to-head priced control is in* P *if prices are represented in unary.*

Proof. We give a proof similar in spirit to the proof of Lemma 4.1. In scoring head-to-head priced control we are given an election $E = (C,V)$, a distinguished candidate d, a preferred candidate $p \in C \setminus \{d\}$, a set of additional voters W and their prices Π. Assume that $\mathrm{diff}_E(d,p) > 0$; otherwise the problem is trivial. Define $\mathrm{gain}_E(v)$ to be the score difference that p gains relative to d if we add voter v to the election E. Let $W' = \{w_1, \dots, w_{|W'|}\}$ be a set of voters from W with positive gain. Let $g(i,p)$ be the maximal total gain, summed over all votes we decided to add, that can be achieved, using first i votes from W' with total price of control not exceeding p. The value $g(i,p)$ can be computed using the following recursive formulation:

$$g(i,p) = \begin{cases} 0 & \text{if } i = 0 \\ g(i-1,p) & \text{if } i \geqslant 1 \text{ and } \Pi(v_i) > p \\ \min\left[g(i-1,p),\ g(i-1, p - \Pi(v_i)) + \mathrm{gain}_E(v_i)\right] & \text{if } i \geqslant 1 \text{ and } \Pi(v_i) \leqslant p \end{cases}$$

Candidate p can tie or beat d if and only if $g(|W'|, K) \geqslant \mathrm{diff}_E(d,p)$, which can be checked in polynomial time using dynamic programming. □

Now we are ready to combine results from Lemma 4.1 and Lemma 4.2 and state the following result.

Theorem 4.3. *Destructive* AV *priced control is in* P *for scoring protocols if either scoring vector entries or prices are represented in unary, and if the scoring vectors for each number of candidates are computable in polynomial time with respect to the required number of candidates*

Proof. In destructive AV priced control problem we are given an election $E = (C, V)$, a distinguished candidate $d \in C$, a set of additional voters W with their prices Π and available budget K. If despised candidate d is already not a unique winner of election E then accept. Otherwise check using a procedure from Lemma 4.1 (when scoring vector entries are represented in unary) or Lemma 4.2 (when prices are represented in unary) if there exists candidate $p \in C \setminus \{d\}$ such that p can tie or beat candidate d after addition of some voters from W within available budget K and accept or reject accordingly. This requires at most $|C| - 1$ executions of algorithms from mentioned theorems, therefore this procedure runs in polynomial time. □

The same approach can be used in destructive control by deleting voters. No significant changes to the above proofs are required. It is simply a matter of updating the definition of gain to reflect that we are deleting voters instead of adding them, and running the dynamic programming over the voters already in the election instead of over those that can be added.

Corollary 4.4. Destructive DV priced control is in P for scoring protocols if either scoring vector entries or prices are represented in unary, and if the scoring vectors for each number of candidates are computable in polynomial time with respect to the required number of candidates.

For most natural classes of scoring protocols, such as, e.g., Borda rule, the assumptions of the above theorems hold. We have the following corollary.

Corollary 4.5. Plurality, veto, k-veto, k-vpproval and Borda count are vulnerable to destructive AV and DV priced control.

As a side comment, we mention that for some rules there is an interesting connection between the complexity of DV-priced-control and the complexity of AV-priced-control. Indeed, for some rules such as Borda, Condorcet, or Copeland, we can reduce the former to the latter.

Theorem 4.6. *Constructive (destructive)* DV *priced control is reducible in polynomial time to constructive (destructive)* AV *priced control for those voting rules for which it is possible to uniquely determine the winners in election E provided that for each two candidates c and d we are given the value $N_E(c, d) - N_E(d, c)$.*

Proof. Let I_{DV} be an instance of constructive (destructive) DV priced control with election $E = (C, V)$, a distinguished candidate $c \in C$, a list of prices associated with voters Π and available budget $K \in \mathbb{N}$. We reduce I_{DV} to instance I_{AV} of constructive (destructive) AV priced control, consisting of:

1. election $E = (C, V)$,
2. distinguished candidate c,
3. set of additional voters W which consists of voters from V with their preference orders reversed, and

4. available budget K.

Further, the price of adding voter $w \in W$ is the same as the price of deleting the voter $v \in V$ (in I_{DV}) from which w was created.

To see why the reduction works, it suffices to make the following simple observation. Let v be some voters in V and let w be the corresponding voter from W. Consider some arbitrary candidates a and b. Since w's preference order is the reverse of that of v, the following holds:

$$N_{(C,V\setminus\{v\})}(a,b) - N_{(C,V\setminus\{v\})}(b,a) = N_{(C,V\cup\{w\})}(a,b) - N_{(C,V\cup\{w\})}(b,a).$$

That is, as far as the values $N_E(a,b) - N_E(b,a)$ are concerned, the effect of deleting voter v is the same as the effect of adding voter w. By the assumption that winners depend only on the values $N_E(a,b) - N_E(b,a)$ (for all the candidates $a, b \in C$), we have that the reduction is correct. Polynomial running time is straightforward. $\square$

To see that the above result applies to Borda, we note that Borda score of a candidate c in election $E = (C, V)$ can be expressed as $\mathrm{score}_E(c) = \sum_{d \in C\setminus\{c\}} N_E(c,d)$ (candidate c receives a point for each candidate d and each voter that ranks c above d). For each $d \in C \setminus \{c\}$, we have that $N_E(c,d) + N_E(d,c) = \|V\|$ and, so, $(N_E(c,d) - N_E(d,c)) + \|V\| = 2N_E(c,d)$. This means that Borda score of candidate c is equal to $\mathrm{score}_E(c) = \frac{1}{2}\sum_{d\in C\setminus\{c\}}(N_E(c,d) - N_E(d,c) + \|V\|)$. So, Borda satisfies the conditions of the above theorem.

Theorem 4.6 is quite interesting since there are relatively few relations known between the complexities of various election related problems. Some similar results were given by Faliszewski, Hemaspaandra, and Hemaspaandra [16] for the case of voter control under k-approval and k-veto, by Hemaspaandra, Hemaspaandra, and Menton [23] for the case of destructive control-by-partition problems, by Faliszewski, Hemaspaandra, and Hemaspaandra [13] for a relation between manipulation and priced bribery, and by Elkind, Faliszewski, and Slinko [10] for the case of the possible winner problem and the swap bribery problem.

4.1.1 Resistance Results

We will now show a scoring protocol for which both destructive priced control by adding voters and destructive priced control by deleting voters are NP-hard. By our previous results, we know that the entries of our scoring protocol cannot be polynomially bounded in the number of candidates.

We design our scoring protocol to facilitate an NP-hardness proof based on a reduction from the X3C (eXact 3 Cover) problem. X3C is a well-known NP-complete problem [21]. We are given a set X and a family $\mathcal{S}$ of three-element subsets of X. We ask if there is an exact cover of X using sets from S. Formally, we define X3C problem as follows:

Name: X3C

Given: Set $X = \{0, \ldots, 3k-1\}$, family $\mathcal{S} = \{S_1, \ldots, S_n\}$ of three-elements subsets of X.

Question: Is there $I \subseteq \{1, \ldots, n\}$ such that $|I| = k$ and $\bigcup_{i\in I} S_i = X$?

We now move on to defining our scoring protocol. For a given positive integer n and three integers $0 \leqslant x < y < z < n$, let $f_n(x, y, z)$ be the number of 3-element subsequences of $\langle n-1, \ldots, 0\rangle$, that are greater or equal to $\langle z, y, x\rangle$ in lexicographical order. For example, if we consider all 3-element subsequences of $\langle 5, \ldots, 0\rangle$, the greatest tuple in lexicographical order is $\langle 5, 4, 3\rangle$, therefore $f_6(3, 4, 5) = 1$. On the other hand $\langle 2, 1, 0\rangle$ is the least subsequence and we have that $f_6(0, 1, 2) = \binom{6}{3} = 20$. More generally, we have that $f_{3k}(3k-3, 3k-2, 3k-1) = 1$ and $f_{3k}(0, 1, 2) = \binom{3k}{3}$.

Definition 4.7. Define scoring protocol $\mathrm{SP_H}$ as follows. If the number of candidates in an election is equal to $m = \binom{3k}{3} + 1$ for some $k \in \mathbb{N}$, then we use scoring vector $\langle \alpha_1, \ldots, \alpha_m\rangle$ such that:

1. $\alpha_{f_{3k}(x,y,z)} = \binom{3k}{3}^x + \binom{3k}{3}^y + \binom{3k}{3}^z$ where $\langle z, y, x\rangle$ is a subsequence of $\langle 3k-1, \ldots, 0\rangle$
2. $\alpha_m = 0$

Otherwise, we use the Borda scoring vector.

Note that if $f_{3k}(a, b, c) < f_{3k}(x, y, z)$, where tuple $\langle z, y, x\rangle$ and tuple $\langle c, b, a\rangle$ are subsequences of $\langle 3k-1, \ldots, 0\rangle$, then $\langle c, b, a\rangle >_{\text{lex}} \langle z, y, x\rangle$. Applying the definition of lexicographical order, we have that $c > z$ or $c = z \wedge b > y$ or $c = z \wedge b = y \wedge a > x$ and in each of these cases it is easy to verify that $\alpha_{f_{3k}(a,b,c)} > \alpha_{f_{3k}(x,y,z)}$. Therefore scoring vectors in $\mathrm{SP_H}$ protocol are monotone.

Before presenting our main resistance results of this section, we give some intuition regarding $\mathrm{SP_H}$ scoring protocol in the following example.

Example 4.8. Let us consider $\mathrm{SP_H}$ scoring vector for the case where the number of candidates is of the form $\binom{3k}{3} + 1$ for some $k \in \mathbb{N}$. The entries of scoring vector, for $k = 2$, are presented on the Figure 6. The entries of our scoring vector, in base $\binom{3k}{3}$ encoding, are either all zeros or are all zeros with three ones. The first important property of $\mathrm{SP_H}$ that we will use in our proofs is a one-to-one correspondence between the entries $\alpha_1, \ldots, \alpha_{\binom{3k}{3}}$ and three element subsets of $\{0, \ldots, 3k-1\}$. The second one is the fact that to cause overflow when adding numbers base $\binom{3k}{3}$ whose digits are only zeros and ones, we need to add at least $\binom{3k}{3}$ numbers. By contrapositive, when we sum fewer than $\binom{3k}{3}$ such numbers, there is no overflow.

Now we can state and prove our main results of this section.

Theorem 4.9. $\mathrm{SP_H}$ *is resistant to destructive* AV*-priced-control.*

Proof. $\mathrm{SP_H}$ is clearly susceptible to control. To show NP-hardness we give a reduction from X3C. Let $(X, \mathcal{S})$ be an X3C instance, where $X = \{0, \ldots, 3k-1\}$, and $\mathcal{S} = \{S_1, \ldots, S_n\}$. We assume that $k < n < \binom{3k}{3}$, as otherwise there is a trivial solution. Let $m = \binom{3k}{3}$. Destructive AV priced control instance is created in the following way:

$\alpha_1 =$ ●●●□□□	$\alpha_6 =$ ●□●□●□	$\alpha_{11} =$ □●●●□□	$\alpha_{16} =$ □●□□●●
$\alpha_2 =$ ●●□●□□	$\alpha_7 =$ ●□●□□●	$\alpha_{12} =$ □●●□●□	$\alpha_{17} =$ □□●●●□
$\alpha_3 =$ ●●□□●□	$\alpha_8 =$ ●□□●●□	$\alpha_{13} =$ □●●□□●	$\alpha_{18} =$ □□●●□●
$\alpha_4 =$ ●●□□□●	$\alpha_9 =$ ●□□●□●	$\alpha_{14} =$ □●□●●□	$\alpha_{19} =$ □□●□●●
$\alpha_5 =$ ●□●●□□	$\alpha_{10} =$ ●□□□●●	$\alpha_{15} =$ □●□●□●	$\alpha_{20} =$ □□□●●●
			$\alpha_{21} =$ □□□□□□

Figure 6: $\mathrm{SP_H}$ scoring vector entries for election with 21 candidates. Values of scoring vector are represented in 20-ary numeral system, ones are depicted by cells with black dot inside, and zeros are depicted by empty cells.

1. Candidate set C consist of $\{d, p\} \cup B$ where $B = \{b_j \mid 1 \leqslant j \leqslant m-1\}$ and d is the despised candidate.
2. The voter set V contains the following voters (when writing $a \succ B$ we mean that $(\forall_{b \in B})[a \succ b]$, and the order among candidates in B is arbitrary unless further specified, similarly for $B \succ a$):
   (a) $(n+k)m^2 + 6m$ voters with preference order $d \succ p \succ B$.
   (b) $(n+k)m^2 + 6m$ voters with preference order $p \succ d \succ B$.
   (c) k voters with preference order $B \cup \{d\} \succ p$, where candidate d is placed in position $f_{3k}(3i-3, 3i-2, 3i-1)$ for i, $1 \leqslant i \leqslant k$
3. The set of W of additional voters consists of one additional vote w_i for each set S_i in $\mathcal{S}$. In vote w_i, candidate d is least preferred, and candidate p is placed in such a way that the addition of w_i to the election increases p's score by $\sum_{j \in S_i} m^j$ (e.g., if $S_i = \{7, 9, 13\}$ then candidate p is placed in position $f_{3k}(7, 9, 13)$). The remaining candidates are placed arbitrarily in the vote. The cost of adding w_i is equal to $\sum_{j \in S_i} m^j$.
4. Total available budget K is equal to $\sum_{i=0}^{3k-1} m^i$.

In election $E = (C, V)$, prior to adding any of the voters from W, the candidates have the following scores:

$$\begin{aligned}
\mathrm{score}_E(d) &= \left[(n+k)m^2 + 6m\right]\left[2m^{3k-1} + 2m^{3k-2} + m^{3k-3} + m^{3k-4}\right] + \sum_{i=0}^{3k-1} m^i \\
\mathrm{score}_E(p) &= \left[(n+k)m^2 + 6m\right]\left[2m^{3k-1} + 2m^{3k-2} + m^{3k-3} + m^{3k-4}\right] \\
\mathrm{score}_E(b_i) &\leqslant \left[(n+k)m^2 + 6m\right]\left[2m^{3k-1} + 2m^{3k-2} + 2m^{3k-5}\right] \\
&\quad + k\left[m^{3k-1} + m^{3k-2} + m^{3k-3}\right]
\end{aligned}$$

Candidate d is the unique winner of this election with score advantage of $\sum_{i=0}^{3k-1} m^i$ (equal to the available budget K) over the candidate p, and score advantage of more than $n(m^{3k-1} + m^{3k-2} + m^{3k-3})$ over candidates in B.

If the input X3C instance has a solution, then adding votes from W that correspond to the sets S_i that constitute an exact cover of X increases the score of candidate p by K and requires budget of K. The score of the despised candidate d remains the same and destructive control is successful.

For the reverse direction, if control is possible then it must be a result of a tie between d and p. This is so because the scores of candidates from B can be increased by no more than $n(m^{3k-1}+m^{3k-2}+m^{3k-3})$, which is not sufficient to tie or beat with d. Moreover, the score of candidate p must be increased by at least $\mathrm{diff}_E(d,p) = K$, and by no more than the available budget K. Thus the sets S_i that correspond to the added voters must form an exact cover of X (recall the second property from Example 4.8). □

Destructive priced control by deleting voters also is NP-hard for our scoring rule.

Theorem 4.10. SP_{H} *is resistant to destructive* DV *priced control.*

Proof. We give a reduction from X3C. Let $(X, \mathcal{S})$ be an X3C instance, where $X = \{0, \dots, 3k-1\}$, and $\mathcal{S} = \{S_1, \dots, S_n\}$. We assume that $k < n < \binom{3k}{3}$ (otherwise there is a trivial solution). Let $m = \binom{3k}{3}$. Destructive priced control by deleting voters instance is created in the following way:

1. Candidate set C consists of $\{d, p\} \cup B$ where $B = \{b_i \mid 1 \leqslant i \leqslant m-1\}$, and d is the despised candidate.

2. The available budget K is $\sum_{i=0}^{3k-1} m^i$.

3. Voters set V contains the following voters:

   (a) $(2n+k)m^2 + 9m$ voters with preference order $d \succ p \succ B$ and cost $K+1$.

   (b) $(2n+k)m^2 + 9m$ voters with preference order $p \succ d \succ B$ and cost $K+1$.

   (c) k voters with preference orders of the form $B \cup \{d\} \succ p$, where candidate d is placed in position $f_{3k}(3i-3, 3i-2, 3i-1)$ for each i in $\{1, \dots, k\}$, cost of each vote is $K+1$

   (d) n voters with preference orders of the form $B \cup \{p\} \succ d$. For each $S_i \in \mathcal{S}$, there is a vote in which p is placed in such a way as to receive score $\sum_{j \in S_i} m^j$ from this vote. The cost of each vote is $K+1$.

   (e) n voters, one for each S_i in $\mathcal{S}$, with preference orders of the form $B \cup \{d\} \succ p$, where in the vote corresponding to set S_i, candidate d is placed in such a way as to receive score of $\sum_{j \in S_i} m^j$. For each i, the cost of the vote corresponding to S_i is $\sum_{j \in S_i} m^j$.

Candidates receive the following scores in election $E = (C, V)$:

$$\begin{aligned}
\mathrm{score}_E(d) &= \Big[(2n+k)m^2 + 9m\Big]\Big[2m^{3k-1} + 2m^{3k-2} + m^{3k-3} + m^{3k-4}\Big] \\
&\quad + \sum_{i=1}^{n}\sum_{j\in S_i} m^j + \sum_{i=0}^{3k-1} m^i \\
\mathrm{score}_E(p) &= \Big[(2n+k)m^2 + 9m\Big]\Big[2m^{3k-1} + 2m^{3k-2} + m^{3k-3} + m^{3k-4}\Big] + \sum_{i=1}^{n}\sum_{j\in S_i} m^j \\
\mathrm{score}_E(b_i) &\leqslant \Big[(2n+k)m^2 + 9m\Big]\Big[2m^{3k-1} + 2m^{3k-2} + 2m^{3k-5}\Big] \\
&\quad + (2n+k)\Big[m^{3k-1} + m^{3k-2} + m^{3k-3}\Big]
\end{aligned}$$

Candidate d is the unique winner of election E with score advantage of $\sum_{i=0}^{3k-1} m^i$ (equal to the available budget K) over p, and with score advantage of more than $n(m^{3k-1}+m^{3k-2}+m^{3k-3})$ over candidates from B.

If input X3C instance has a solution, then deleting votes from V of type (3e) that correspond to sets S_i that constitute exact cover of X, decreases the score d by K and requires budget of K. The score of candidate p is unchanged, and thus p and d tie and, so, destructive control is successful.

In the other direction, if control is possible it must be a result of a tie between d and p. This is so because existing score differences between candidates in B and d is higher than a change that could be introduced by deleting votes of type (3e) (and no other types of votes can be removed from the election). The score of candidate d must be therefore decreased by at least K to ensure d ties or loses with p. But at the same time it must not exceed K, to be within budget, as the introduced score difference between d and p is equal to the price of the control action. Thus the total price must be exactly K and the deleted voters directly correspond to sets S_i constituting exact cover of X. □

We believe that the above results are quite intriguing. While our scoring protocol $\mathrm{SP_H}$ is not likely to be used in any real-life election, it also is not completely unnatural. It is interesting if one can show that destructive control by adding/deleting voters is NP-hard for scoring vectors of the form $\langle 2^{m-1}, 2^{m-2}, \ldots, 2^1, 2^0\rangle$. We leave this as an interesting open problem.

5 Summary

In this work we examined the computational complexity of election control for the case where different control actions (such as adding/deleting different candidates or voters) may come at different prices. We argued that such problems are useful ways of modeling problems that arise in planning political campaigns.

We examined plurality, approval, Condorcet, and Copeland rules and we have shown that introducing prices does not affect the complexity of control problems for these rules.

On the other hand, we have shown that there are scoring protocols for which unpriced destructive control is polynomial-time solvable, but for which introducing prices moves the problem to be NP-hard. This is particularly interesting when we compare the complexity of priced control with the complexity of unpriced control, for weighted elections. For the latter, Faliszewski, Hemaspaandra, and Hemaspaandra [16] argue that destructive voter control is polynomial-time solvable for all scoring protocols. We have shown that this is not the case for priced control.

Our work opens several interesting research directions. First, one could seek if there are natural voting rules for which introducing prices increases the complexity of control problems. It is also interesting to consider the complexity of priced control in restricted domains, such as the single-peaked domain or the single-crossing domains. Another potential research direction is to consider approximation algorithms for the priced control problems.

References

[1] J. Bartholdi, III, C. Tovey, and M. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(2):157–165, 1989.

[2] 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.

[3] D. Baumeister, G. Erdélyi, O. Erdélyi, and J. Rothe. Control in judgment aggregation. In *Proceedings of the 6th European Starting AI Researcher Symposium*, pages 23–34. IOS Press, August 2012.

[4] D. Baumeister, M. Roos, J. Rothe, L. Schend, and L. Xia. The possible winner problem with uncertain weights. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 133–138. IOS Press, August 2012.

[5] N. Betzler and J. Uhlmann. Parameterized complexity of candidate control in elections and related digraph problems. *Theoretical Computer Science*, 410(52):43–53, 2009.

[6] S. Brams and P. Fishburn. Voting procedures. In K. Arrow, A. Sen, and K. Suzumura, editors, *Handbook of Social Choice and Welfare, Volume 1*, pages 173–236. Elsevier, 2002.

[7] F. Brandt, M. Brill, E. Hemaspaandra, and L. Hemaspaandra. Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. In *Proceedings of AAAI-10*, pages 715–722. AAAI Press, July 2010.

[8] Y. Chevaleyre, J. Lang, N. Maudet, J. Monnot, and L. Xia. New candidates welcome! possible winners with respect to the addition of new candidates. *Mathematical Social Sciences*, 64(1):74–88, 2012.

[9] E. Elkind and P. Faliszewski. Approximation algorithms for campaign management. In *Proceedings of WINE-10*, pages 473–482. Springer-Verlag *Lecture Notes in Computer Science #6484*, December 2010.

[10] E. Elkind, P. Faliszewski, and A. Slinko. Swap bribery. In *Proceedings of SAGT-09*, pages 299–310. Springer-Verlag *Lecture Notes in Computer Science #5814*, October 2009.

[11] E. Elkind, P. Faliszewski, and A. Slinko. Cloning in elections: Finding the possible winners. *Journal of Artificial Intelligence Research*, 42:529–573, 2011.

[12] G. Erdélyi, M. Fellows, J. Rothe, and L. Schend. Control complexity in Bucklin and fallback voting. Technical Report arXiv:1103.2230 [cs.CC], Computing Research Repository, arXiv.org/corr/, August 2012. March, 2011. Revised August, 2012.

[13] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. How hard is bribery in elections? *Journal of Artificial Intelligence Research*, 35:485–532, 2009.

[14] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. Using complexity to protect elections. *Communications of the ACM*, 53(11):74–82, 2010.

[15] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. Multimode control attacks on elections. *Journal of Artificial Intelligence Research*, 40:305–351, 2011.

[16] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra. Weighted electoral control. In *Proceedings of AAMAS-13*, pages 367–374. International Foundation for Autonomous Agents and Multiagent Systems, May 2013.

[17] 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.

[18] 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.

[19] P. Faliszewski and A. Procaccia. AI's war on manipulation: Are we winning? *AI Magazine*, 31(4):52–64, 2010.

[20] H. Garcia-Molina. Elections in a distributed computing system. *IEEE Transactions on Computers*, C-31(1):48–59, 1982.

[21] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[22] A. Gibbard. Manipulation of voting schemes. *Econometrica*, 41(4):587–601, 1973.

[23] E. Hemaspaandra, L. Hemaspaandra, and C. Menton. Search versus decision for election manipulation problems. In *Proceedings of the 30th Annual Symposium on Theoretical Aspects of Computer Science*, pages 377–388. Leibniz-Zentrum für Informatik, 2013.

[24] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP. *Journal of the ACM*, 44(6):806–825, 1997.

[25] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, 171(5–6):255–285, 2007.

[26] E. Hemaspaandra, H. Spakowski, and J. Vogel. The complexity of Kemeny elections. *Theoretical Computer Science*, 349(3):382–391, 2005.

[27] M. Kuta and J. Kitowski. Benchmarking high performance architectures with natural language processing algorithms. *Computer Science*, 12:19–31, 2011.

[28] L. Lamport. Paxos made simple. *SIGACT News*, 32(4):51–58, December 2001.

[29] 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.

[30] H. Liu and D. Zhu. Parameterized complexity of control problems in maximin election. *Information Processing Letters*, 110(10):383–388, 2010.

[31] 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.

[32] D. Parkes and L. Xia. A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs. In *Proceedings of AAAI-12*, pages 1429–1435, July 2012.

[33] J. Rothe, H. Spakowski, and J. Vogel. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems*, 36(4):375–386, 2003.

[34] M. Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2):187–217, 1975.

[35] K. Wojtas and P. Faliszewski. Possible winners in noisy elections. In *Proceedings of AAAI-12*, pages 1499–1505, July 2012.

[36] L. Xia. Fixed-parameter tractability of integer generalized scoring rules. In *Proceedings of AAMAS-14*, pages 1599–1600, 2014.