ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230338

Using Weighted Matching to Solve 2-Approval/Veto Control and Bribery

Zack Fitzsimmons[a;*] **and Edith Hemaspaandra**[b]

[a]Department of Mathematics and Computer Science, College of the Holy Cross, Worcester, MA 01610 USA
[b]Department of Computer Science, Rochester Institute of Technology, Rochester, NY 14623 USA

Abstract. Determining the complexity of election attack problems is a major research direction in the computational study of voting problems. The paper "Towards completing the puzzle: complexity of control by replacing, adding, and deleting candidates or voters" by Erdélyi et al. (JAAMAS 2021) provides a comprehensive study of the complexity of control problems. The sole open problem is constructive control by replacing voters for 2-Approval. We show that this case is in P, strengthening the recent RP (randomized polynomial-time) upper bound due to Fitzsimmons and Hemaspaandra (IJCAI 2022). We show this by transforming 2-Approval CCRV to weighted matching. We also use this approach to show that priced bribery for 2-Veto elections is in P. With this result, and the accompanying (unsurprising) result that priced bribery for 3-Veto elections is NP-complete, this settles the complexity for k-Approval and k-Veto standard control and bribery cases.

1 Introduction

Elections are the most widely-used way to aggregate the preferences of a group of agents over a common set of alternatives. Applications include political elections as well as multiagent systems in artificial intelligence applications. Thus it is important to study computational aspects of elections including problems such as winner determination and different ways of strategically affecting the outcome, which are referred to as election-attack problems (see, e.g., Brandt et al. [3]).

Electoral control and bribery are two important examples of election-attack problems. Control models the actions of an election chair who has control over the structure of the election (e.g., the set of voters) and modifies this structure to ensure a preferred outcome (e.g., by deleting voters) [1]. Different control actions model real-world settings such as get-out-the-vote drives or adding "spoiler" candidates to an election to a preferred candidate wins. The standard control cases of adding or deleting voters or candidates can be naturally extended to replacing voters or candidates, which for example models settings where the size of the collection of voters is fixed to its initial amount (e.g., in a parliament) and so the control action must work within this restriction [20]. Bribery considers how a strategic agent can set the votes of a subcollection of the voters to ensure a preferred outcome [9]. The computational study of control and bribery is a major research direction in computational social choice (see Faliszewski and Rothe [11]).

Asking voters to state their most preferred candidate where candidates with the highest number of approvals win is one of the most natural ways to elicit preferences to reach a group decision. This is referred to as the Plurality rule. We consider k-Approval, which generalizes this for fixed k. We additionally consider k-Veto where each voter vetoes their k least preferred candidates, and candidates with the fewest vetoes win.

The study of electoral control has led to many different papers considering the complexity of different actions for important voting rules. A recent comprehensive study on the complexity of control by Erdélyi et al. [8] sought to settle the complexity of the open problems that have remained for control by replacing, adding, and deleting candidates or voters. In the large number of cases summarized and completed in this work only the complexity of control by replacing voters for 2-Approval remained open. In very recent work by Fitzsimmons and Hemaspaandra [13] this problem was shown to be in RP (randomized polynomial-time). We strengthen this result and show this problem in P, thus completing the goal of the comprehensive work by Erdélyi et al. [8].

The complexity of standard control and bribery for k-Approval and k-Veto elections was systematically studied by Lin [18, 19], which left only a few open cases. Several of these open cases concern weighted voting and were handled by Faliszewski, Hemaspaandra, and Hemaspaandra [10]. The last remaining open cases were 2-Approval and 2/3-Veto priced bribery, where setting the votes of different voters may incur different costs. Bredereck et al. [4] state that the 2-Approval case is in P in the summary for a Dagstuhl working group. The 2-Veto case was incorrectly claimed NP-complete by Bredereck and Talmon [5] and their classification of the 3-Veto case relied on the same incorrect proof. We resolve these issues in the literature in the present paper.

We summarize the main contributions of our paper below.

- We show that (constructive) control by replacing voters for 2-Approval, the sole remaining open case from Erdélyi et al. [8], that was very recently shown to be in RP (randomized polynomial time) by Fitzsimmons and Hemaspaandra [13], is in fact in P (Theorem 3).
- We also settle the complexity of the last remaining cases from the study of k-Approval and k-Veto by Lin [18, 19] by showing that priced bribery for 2-Veto is in P (Theorem 4), while priced bribery for 3-Veto is NP-complete (Theorem 5).
- To prove our polynomial-time results we use transformations from voting problems to weighted matching. Weighted matching has been seldom used in the computational study of voting problems and our nontrivial polynomial-time results illustrate its usefulness.

* Corresponding Author. Email: zfitzsim@holycross.edu.

2 Preliminaries

An election (C, V) consists of a set of candidates C and a collection of voters V where each voter has a corresponding strict total order ranking of the candidates in C, referred to as a vote. A voting rule is a mapping from an election to a subset of the candidate set referred to as the winners.

Our results concern the k-Approval and k-Veto voting rules. In a k-Approval election each candidate receives one point from each voter who ranks them among their top-k preferred candidates and candidates with the highest score win. In a k-Veto election each candidate receives one veto from each voter who ranks them among their bottom-k preferred candidates and candidates with the fewest vetoes win.

2.1 Election-Attack Problems

As explained in the introduction, electoral control and bribery are important and well-studied families of election-attack problems. To solve important open questions in the literature we consider control by replacing voters and priced bribery.

Control by replacing voters [20] models scenarios where the election chair with control over the structure of the election replaces a subcollection of the voters with the same number of unregistered voters to ensure a preferred candidate wins. This model ensures that the size of the collection of voters in the election is the same after the control action. We formally define this problem below.

Name: $\mathcal{R}$-Constructive Control by Replacing Voters (CCRV) [20]

Given: Given an election (C, V), a collection of unregistered voters W, preferred candidate p, and integer k such that $0 \leq k \leq \|V\|$.

Question: Do there exist subcollections $V' \subseteq V$ and $W' \subseteq W$ such that $\|V'\| = \|W'\| \leq k$ and p is a winner of the election $(C, (V - V') \cup W')$ using voting rule $\mathcal{R}$?

Bribery, introduced by Faliszewski, Hemaspaandra, and Hemaspaandra [9], models the actions of an agent who sets the votes of a subcollection of the voters to ensure a preferred candidate wins [9]. We mention that bribery can be thought of as modeling the campaign costs of an election where an agent must determine if there is a subcollection of voters that can be convinced to change their vote to ensure a preferred candidate wins. We consider the problem of priced bribery, which we formally define below.

Name: $\mathcal{R}$-\$Bribery [9]

Given: Given an election (C, V) where each voter $v \in V$ has (integer) price $\pi(v) \geq 0$, preferred candidate p, and budget $k \geq 0$.

Question: Is there a way to change the votes of a subcollection of voters $V' \subseteq V$ within the budget (i.e., $\pi(V') = \sum_{v \in V'} \pi(v) \leq k$) such that p is a winner using voting rule $\mathcal{R}$?

As is standard we look at the nonunique-winner model. Erdélyi et al. [8] remark that their proofs can be modified to work for the unique-winner model, and we remark that our proofs can be easily modified to work for the unique-winner model as well.

2.2 Matching

For our polynomial-time algorithms, we will reduce to the following polynomial-time computable weighted matching problems (we will be writing V_G for sets of vertices to distinguish them from collections V of voters).[1]

Name: Max-Weight Perfect b-Matching for Multigraphs

Given: An edge-weighted multigraph $G = (V_G, E)$, where each edge has nonnegative integer weight, a function $b : V_G \to \mathbb{N}$, and integer $k \geq 0$.

Question: Does G have a perfect b-matching of weight at least k, i.e., does there exist a set of edges $E' \subseteq E$ of weight at least k such that each vertex $v \in V_G$ is incident to exactly $b(v)$ edges in E'?

We define Min-Weight Perfect b-Matching for Multigraphs analogously.

The analogues for simple graphs, denoted by Max-Weight Perfect b-Matching and Min-Weight Perfect b-Matching, are in P [7], and it is easy to reduce Max[Min]-Weight Perfect b-Matching for Multigraphs to Max[Min]-Weight Perfect Matching (Lemma 1) by generalizing the reduction from (unweighted) Perfect b-Matching for Multigraphs to Perfect Matching using the construction from Tutte [24] (see, e.g., Berge [2, Chapter 8]). Lemma 1 shows this for Max-Weight. The case for Min-Weight is analogous.

Lemma 1. *Max-Weight Perfect b-Matching for Multigraphs reduces to Max-Weight Perfect Matching.*

Proof. It is straightforward to reduce Perfect b-Matching for Multigraphs to Perfect Matching using the construction from Tutte [24] (see, for example, Berge [2, Chapter 8]). Given multigraph G and capacity function b, replace each vertex v by a complete bipartite graph $(P_v, \{v_1, \ldots, v_{\delta(v)}\})$, where P_v is a set of $\delta(v) - b(v)$ padding vertices (we assume wlog that $\delta(v) \geq b(v)$, otherwise there is no matching). We number the edges incident with v and for each edge e in G, if this edge is the ith v edge and the jth w edge in G, we have an edge (v_i, w_j).

The same reduction works for the weighted version. We set the weight of (v_i, w_j) to the weight of e and we set the weight of the padding edges to 0. Call the resulting graph G'. It is easy to see that a perfect b-matching of G of weight k will give a perfect matching of G' with weight k: Simply take all the edges corresponding to the matching of G and add padding edges to make this a perfect matching. Since padding edges have weight 0, the obtained perfect matching of G' has weight k. For the converse, note that any perfect matching of G' consists of padding edges plus a set of edges corresponding to a perfect b-matching of G. □

From the above lemma we have the following.

Theorem 2. *Max-Weight and Min-Weight Perfect b-Matching for Multigraphs are in* P.

[1] Though many matching problems are in P (see, e.g., [15], Section 7), one has to be careful since minor-seeming variations can make these problems hard. In the context of voting problems, the weighted version of control by adding voters for 2-Approval where we limit the number of voters to add can be viewed as a variation of a weighted matching problem [18]. Since weighted control by adding voters for 2-Approval is NP-complete [10], so is the corresponding matching problem. We also mention here that Exact Perfect Matching (which asks if a graph where a subset of the edges is colored red has a perfect matching with exactly a given number of red edges) introduced by Papadimitriou and Yannakakis [23] and shown to be in RP by Mulmuley, Vazirani, and Vazirani [22] is not known to be in P.

3 2-Approval-CCRV is in P

In the paper "Towards Completing the Puzzle: Complexity of Control by Replacing, Adding, and Deleting Candidates or Voters" [8], the sole open problem is the complexity of constructive control by replacing voters for 2-Approval. Very recently, this problem was shown to be in RP (randomized polynomial time) [13]. We now strengthen this upper bound to show the problem in P.

Theorem 3. *2-Approval-CCRV is in* P.

Proof. We will reduce constructive control by replacing voters for 2-Approval to Max-Weight Perfect b-Matching for Multigraphs, which is defined in the preliminaries.

Since Max-Weight Perfect b-Matching for Multigraphs is in P (Theorem 2), this immediately implies that control by replacing voters for 2-Approval is in P.

Consider an instance of 2-Approval-CCRV with candidates C, registered voters V, unregistered voters W, preferred candidate p, and an integer $k \geq 0$. Let n be the size of V. We ask if it is possible to make p a winner by replacing at most k voters from V by a subcollection of W of the same size. We rephrase our question as follows: Does there exist a subcollection $\widehat{V}$ of $V \cup W$ of size n such that p is a winner and such that $\widehat{V}$ contains at least $n - k$ voters from V?

Note that we can assume that we will put as many voters approving p in $\widehat{V}$ as possible. Let V_p be the subcollection of V approving p and let W_p be the subcollection of W approving p. We put all of V_p in $\widehat{V}$ and as many voters of W_p as possible, keeping in mind that $\widehat{V}$ should contain at most k voters from W and exactly n voters total. This fixes fs_p, the final score of p.[2]

$$fs_p = \|V_p\| + \min(k, n - \|V_p\|, \|W_p\|)$$

And we re-rephrase our question as follows: Does there exist a subcollection $\widehat{V}$ of $V \cup W$ such that

- for each candidate c, the score of c is at most fs_p,
- $\widehat{V}$ contains n voters,
- $\widehat{V}$ contains at least $n - k$ voters from V,
- the score of p is fs_p.

Example 1. *Consider the instance of 2-Approval-CCRV with candidates $\{a, b, c, p\}$, preferred candidate p, $k = 3$, and registered and unregistered voters V and W, respectively.*

V consists of $n = 5$ voters:

- *One approving b and c.*
- *Two approving a and b.*
- *Two approving p and b.*

W consists of four voters:

- *Two approving a and c.*
- *One approving b and c.*
- *One approving p and a.*

Initially the scores of p, a, b, and c are 2, 2, 5, and 1, respectively. Note that

$$\begin{aligned} fs_p &= \|V_p\| + \min(k, n - \|V_p\|, \|W_p\|) \\ &= 2 + \min(3, 3, 1) \\ &= 3. \end{aligned}$$

$\widehat{V}$ consists of the following $n = 5$ voters.

- *The two voters from V approving p and b (note that $2 \geq n - k$).*
- *Three voters from W: two approving a and c; one approving p and a.*

Notice that for the election after control, with voters $\widehat{V}$, the score of each candidate is at most $fs_p = 3$ and p is a winner with score $fs_p = 3$.

In our reduction, a voter in $V \cup W$ approving a and b will correspond to an (undirected) edge (a, b). We will call such an edge a "voter-edge." (We will also have padding edges to make everything work.)

In the graph corresponding to the election, $\widehat{V}$ will correspond to a matching (which will be padded to a perfect matching with padding edges) and the score of a candidate c in $\widehat{V}$ will be the number of edges corresponding to $\widehat{V}$ incident with vertex c. We set $b(c)$ to fs_p, which will ensure that the number of edges corresponding to $\widehat{V}$ incident with c is at most $b(c)$.

We still need to ensure the following in our graph.

- There are exactly n voter-edges in the matching.
- There are at least $n - k$ voter-edges corresponding to voters in V in the matching.
- There are exactly fs_p voter-edges in the matching that are incident with p (this ensures that the final score of p is fs_p as advertised).

Recall from Footnote 1 that for example Exact Perfect Bipartite Matching is not known to be in P (and in fact the weighted version is even NP-complete since Subset Sum straightforwardly reduces to it [23]), and so care needs to be taken with handling "exactness" restrictions in a matching. This is where we will be using that we are reducing to a *perfect* matching problem. We will add an extra vertex x that will take up the slack, i.e., it will ensure that the perfect matching contains exactly n voter-edges. Note that since for every candidate c, $b(c) = fs_p$, and we want exactly n voter-edges in the matching, we have a total of $\|C\| fs_p - 2n$ amount of vertex capacity left to cover.

We now formally define our graph G:

Vertices $V(G) = C \cup \{x\}$.

- For each $c \in C$, $b(c) = fs_p$.
- $b(x) = \|C\| fs_p - 2n$. If this is negative, control is not possible.

Edges

- For each voter in $V \cup W$ that approves a and b, add a voter-edge (a, b).
- For each candidate $c \in C - \{p\}$, add $b(c) = fs_p$ padding edges (c, x).

Weights A perfect matching will saturate x (i.e., contain $b(x) = \|C\| fs_p - 2n$ edges incident with x) and will contain exactly n voter-edges. We want to maximize the number of voter-edges corresponding to voters in V, and so we set the weight of those voter-edges to 1 and the weight of all other edges (i.e., the voter-edges corresponding to W and the padding edges) to 0.

We now prove that our reduction is correct:

We will show that p can be made a winner by replacing at most k voters if and only if G has a matching of weight at least $n - k$. See Figure 1 for an example of this reduction applied to the instance of 2-Approval-CCRV from Example 1.

[2] We are not claiming that fs_p is the final score of p for every successful control action; however, if there is a successful control action, then there is a successful control action in which the final score of p is fs_p and so we can restrict ourselves to control actions where the final score of p is fs_p.

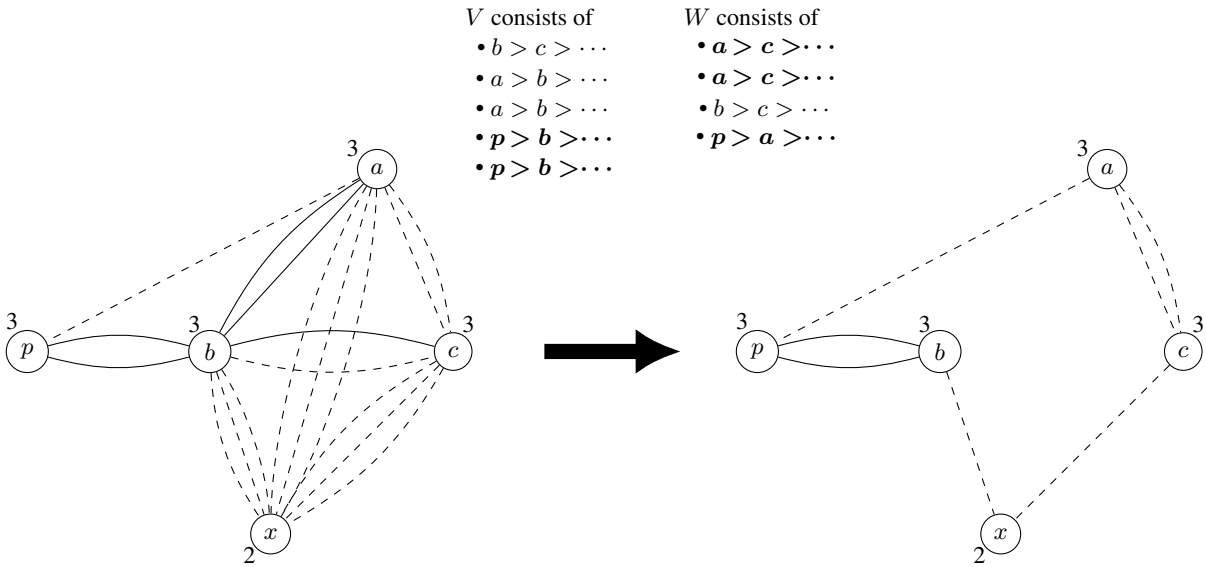

Figure 1. Example of the construction from the proof of Theorem 3 applied to the instance of 2-Approval-CCRV from Example 1 (left) and the perfect b-matching of weight $\geq n - k = 2$ corresponding to $\widehat{V}$ (right). The collections of voters V and W are given with the voters in $\widehat{V}$ in bold. In the graph solid edges have weight 1, dashed edges have weight 0, and each vertex is labeled with its b-value.

If p can be made a winner by replacing at most k voters, then there exists a subcollection $\widehat{V}$ of $V \cup W$ of size n such that p is a winner in $\widehat{V}$, p's score is fs_p, and such that $\widehat{V}$ contains at least $n - k$ voters from V. Then the edges corresponding to $\widehat{V}$ are a matching of G, since the number of edges incident with a candidate c is the score of c in $\widehat{V}$, which is at most $fs_p = b(c)$. And the weight of this matching is at least $n - k$. Since there are $fs_p = b(p)$ edges incident with p in the matching, we can add $b(x)$ edges between x and vertices in $C - \{p\}$ to the matching, to obtain a perfect matching of weight at least $n - k$.

For the converse, suppose G has a perfect matching of weight at least $n-k$. Since the matching is perfect, it contains exactly n voter-edges and at least $n-k$ of these correspond to voters in V. Since the matching is perfect, vertex p is incident to exactly fs_p edges in the matching and all of these are voter-edges. Let $\widehat{V}$ be the collection of voters corresponding to voter-edges in the matching. It is immediate that $\widehat{V}$ is of size n, that $\widehat{V}$ contains at least $n - k$ voters from V, and that p is a winner with score fs_p. □

4 2-Veto-$Bribery

We now turn to 2-Veto-$Bribery and show that this problem is in P. Lin [19, Proof of Theorem 3.8.2] showed that Min-Weight b-Cover for Multigraphs easily reduces to 2-Veto-$Bribery, which implies that 2-Veto-$Bribery is unlikely to have a simple polynomial-time algorithm.

In fact, 2-Veto-$Bribery was incorrectly claimed to be NP-complete by Bredereck and Talmon [5]. See the full version of our paper [14] for details on why their proof is incorrect.

Theorem 4. *2-Veto-$Bribery is in* P.

Proof. Consider an instance of 2-Veto-$Bribery with candidates C, voters V, preferred candidate p, and budget $k \geq 0$. For v a voter, we denote the price of v by $\pi(v)$. If there are at most two candidates the problem is trivial and so we assume we have at least three candidates. We will be using that Min-Weight Perfect b-matching for Multigraphs is in P (Theorem 2). In our reduction we will let a voter v who vetoes a and b correspond to an edge (a, b) of weight $\pi(v)$, the price of v. We will call such an edge a voter-edge. We will argue about 2-Veto elections in terms of the number of vetoes each candidate gets (this way each voter affects only two candidates and these numbers are nonnegative). Our bribery instance is positive if and only if there is a bribery within budget k such that after bribery the number of vetoes for p is *at most* the number of vetoes for c for every other candidate c.

Let V_p be the collection of voters in V that veto p and let V'_p be the collection of voters in V that do not veto p, i.e., $V'_p = V - V_p$. For each ℓ_p, ℓ'_p such that $0 \leq \ell_p \leq \|V_p\|$ and $0 \leq \ell'_p \leq \|V'_p\|$ we determine whether there exists a successful bribery where ℓ_p voters in V_p and ℓ'_p voters in V'_p are bribed. Note there are polynomially many such pairs ℓ_p, ℓ'_p. Let fv_p be the final number of vetoes for p (i.e., after bribery). Without loss of generality, we assume that we never bribe a voter to veto p. So $fv_p = v_p - \ell_p$, where v_p is the number of vetoes for p in V. After bribery, we want each candidate $c \neq p$ to have at least fv_p vetoes. We already mentioned that each voter v that vetoes a and b will correspond to an edge (a, b) of weight $\pi(v)$. We want the collection of voters we bribe to correspond to a matching, so if an edge (a, b) is in the matching, that corresponds to the corresponding voter—who vetoes a and b—being bribed, and so the number of vetoes for a and b decreases by 1. The obvious thing to do would be to set $b(c)$ to the number of vetoes for c that can be deleted, i.e., $b(c) = v_c - fv_p$, where v_c is the number of vetoes for c in V. However, things are much more complicated here, since we need to ensure that our matching contains exactly ℓ_p voter-edges corresponding to voters in V_p and exactly ℓ'_p voter-edges corresponding to voters in V'_p, and that the $\ell_p + \ell'_p$ bribed voters each veto two candidates.

At first glance it may seem that we should just simply bribe only voters that veto p and so ℓ'_p would be unnecessary. However, since our voters have prices this is not the case. The most obvious example

would be when all voters vetoing p have a price greater than the budget and p is not initially a winner. Additionally, it is not difficult to construct scenarios where bribery is possible when voters not vetoing p are bribed and it is not possible for p to be made a winner by bribing only voters vetoing p (and the prices are each within the budget). See Example 2 below, which we will also use in Figure 2 to demonstrate our reduction7.

Example 2. *Consider the instance of 2-Veto-$Bribery with candidates $\{a, b, c, p\}$, preferred candidate p, budget $k = 3$, and V consists of the following voters.*

- *One voter vetoing a and b with price 1.*
- *Two voters vetoing b and c with price 1.*
- *Two voters vetoing b and p with price 1.*
- *Four voters vetoing c and p with price 2.*

From the voters in V, $v_a = 1$, $v_b = 5$, $v_c = 6$, and $v_p = 6$.

There is a bribery where p wins where $\ell_p = 2$ and $\ell'_p = 1$ (i.e., that bribes two voters that veto p and one that does not). This sets $fv_p = 6 - 2 = 4$. Bribe the two voters vetoing b and p and one of the voters vetoing b and c to each veto a and b. After bribery a and p have 4 vetoes, b and c have 5 vetoes, and so p is a winner.

We rephrase our problem until we are in a matching-friendly form. Bribery is possible if and only if there exists a subcollection of voters $\widehat{V} \subseteq V$ (the collection of voters we bribe) such that:

1. $\widehat{V}$ contains exactly ℓ_p voters of V_p (voters vetoing p),
2. $\widehat{V}$ contains exactly ℓ'_p voters of V'_p (voters not vetoing p),
3. $\pi(\widehat{V}) \leq k$, and
4. there exists a collection of $\ell_p + \ell'_p$ votes (the votes of the bribed voters after bribery) such that for all c, the number of vetoes for c after bribery is at least fv_p.

Writing v_c for the number of vetoes for c in V and $\widehat{v}_c$ for the number of vetoes for c in $\widehat{V}$, we can rewrite the fourth requirement above as follows.

4. There exist integers bv_c for each $c \in C - \{p\}$ (these will be the number of vetoes that the bribery gives to c), such that
   - $v_c - \widehat{v}_c + bv_c$ (the number of vetoes for c after bribery) $\geq fv_p$,
   - $0 \leq bv_c \leq \ell_p + \ell'_p$ (each bribed voter can give only one veto to c), and
   - $\sum_{c \in C - \{p\}} bv_c = 2(\ell_p + \ell'_p)$ (the bribery gives $2(\ell_p + \ell'_p)$ vetoes total to candidates $C - \{p\}$).

Note that the maximum number of vetoes for candidate c after bribery is $v_c + \ell_p + \ell'_p$ (this happens when we do not bribe voters that veto c and all bribed voters are bribed to veto c). If this quantity is less than fv_p for some c, bribery is not possible for this choice of ℓ_p and ℓ'_p. Otherwise, $b(c) = v_c + \ell_p + \ell'_p - fv_p$ is the number of vetoes c can lose from the maximum number of possible vetoes for c.

We now define our graph G:

Recall that we are reducing to Min-Weight Perfect b-Matching for Multigraphs and that we have at least three candidates. An example of this construction applied to the instance of 2-Veto-$Bribery from Example 2 is presented in Figure 2. Note that in order for G to have a b-matching, all b-values must be nonnegative.

- $V(G) = C \cup \{x, y\}$. (x will ensure that the perfect matching contains exactly ℓ'_p edges from V'_p; this is similar to the role of x in the construction for 2-Approval-CCRV. y will handle the vetoes given by the bribery.)
- For each voter in $v \in V$ vetoing a and b, add a voter-edge (a, b) with weight $\pi(v)$.
- $b(p) = \ell_p$. This will ensure that a perfect matching contains exactly ℓ_p voter-edges corresponding to voters that veto p.
- For each $c \in C - \{p\}$, $b(c) = v_c + \ell_p + \ell'_p - fv_p$.
- For each $c \in C - \{p\}$, add $(\ell_p + \ell'_p)$ bribe-edges (c, y) with weight 0. This is the maximum number of vetoes that c can receive from the bribery. The bribe-edges *not* in the perfect matching will correspond to the vetoes that the bribery gives to c.[3] Since the bribery gives a total of $2(\ell_p + \ell'_p)$ vetoes and we have a total of $\|C - \{p\}\|(\ell_p + \ell'_p)$ bribe-edges, we set
$$b(y) = (\|C\| - 3)(\ell_p + \ell'_p).$$
- For each $c \in C - \{p\}$, add $b(c)$ padding edges (c, x) of weight 0. We want to ensure that a perfect matching contains exactly ℓ'_p voter-edges corresponding to voters that do not veto p. The excess of b-values is
  - $\sum_{c \in C - \{p\}} b(c)$
  - minus ℓ_p (one for each of the ℓ_p voter-edges incident with p; note that each such edge is incident with one $c \in C - \{p\}$)
  - minus $2\ell'_p$ (two for each of the ℓ'_p voter-edges not incident with p)
  - minus $b(y)$ (the number of bribe-edges in the perfect matching; note that each such edge is incident with one $c \in C - \{p\}$).

  We set $b(x)$ to this value.

We now prove that our reduction is correct:

We need to show that there is a successful bribery that bribes ℓ_p voters from V_p and ℓ'_p voters from V'_p of cost $\leq k$ if and only if G has perfect b-matching of weight $\leq k$. Note that G is independent of k, and so it suffices to prove the following.

> For all k there is a successful bribery that bribes ℓ_p voters from V_p and ℓ'_p voters from V'_p of cost k if and only if G has a perfect b-matching of weight k.

If there is a successful bribery of cost k that bribes ℓ_p voters from V_p and ℓ'_p voters from V'_p, let $\widehat{V}$ be the collection of bribed voters, and let fv_p, v_c, and $\hat{v}_c$ be as defined above. For $c \in C - \{p\}$, let bv_c be the number of vetoes the bribery gives to c. Without loss of generality, assume that the bribery does not give vetoes to p. Recall that we have at least three candidates.

We will show that the following set of edges E' is a perfect b-matching of G of weight k.

- the voter-edges corresponding to $\widehat{V}$ (since these are the only nonzero weight edges, the matching will have total weight $\pi(\widehat{V}) = k$),
- for each $c \in C - \{p\}$, $(\ell_p + \ell'_p) - bv_c$ bribe-edges (c, y) representing the vetoes not given to c by the bribery,

[3] At this point one might wonder why we are having the *complement* of the matching correspond to the final election rather than the matching itself. The reason is that we would then be arguing about covers (instead of matchings) which would have to be padded to perfect covers. Padding covers is a little harder than padding matchings; with matchings we just add edges, but with covers we also need to update the b-values.

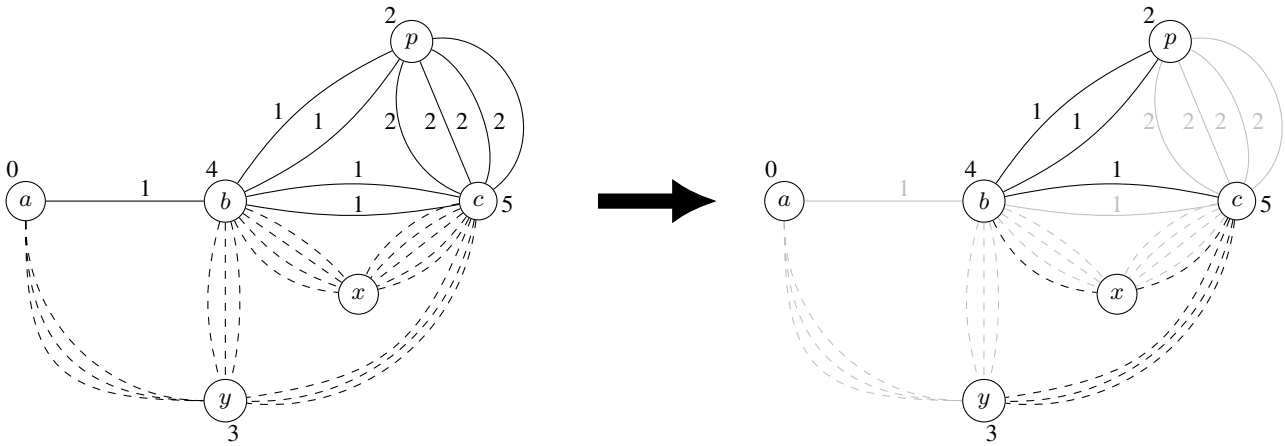

Figure 2. Example of the construction from the proof of Theorem 4 applied to the instance of 2-Veto-$Bribery from Example 2 (left) and the perfect b-matching of weight $\leq k = 3$ corresponding to $\widehat{V}$ (right). In the graph solid edges are labeled with their corresponding weight, dashed edges have weight 0, and each vertex is labeled with its b-value. Recall that a perfect matching corresponds to the complement of the final bribed election. The "grayed out" edges on the right show the final bribed election, which has four voters vetoing p and c, one voter vetoing a and b, one voter vetoing b and c, and three bribe vetoes for a and three bribes vetoes for b (corresponding to the three (a, y) edges and the three (b, y) edges).

- for each $c \in C - \{p\}$, padding edges (c, x) to meet the corresponding b-value. Since c is incident with $\widehat{v}_c$ voter-edges and $(\ell_p + \ell'_p) - bv_c$ bribe-edges, we need $b(c) - \widehat{v}_c - ((\ell_p + \ell'_p) - bv_c)$ padding edges.

We now show that E' is a perfect b-matching of G. Part of showing this is making sure that all the numbers of edges are nonnegative.

- The edges in E' incident with p are exactly the ℓ_p voter-edges in $\widehat{V}$ that veto p and $b(p) = \ell_p \geq 0$.
- For $c \in C - \{p\}$, the definition ensures that the number of voter-edges + bribe-edges + padding edges in E' incident with c add up to $b(c)$. But we have to ensure that these numbers are all nonnegative. This is immediate for the number of voter-edges incident with c and for the number of bribe-edges incident with c, and so it remains to show that the number of padding edges incident with c, $b(c) - \hat{v}_c - ((\ell_p + \ell'_p) - bv_c)$, is nonnegative. Since $b(c) = v_c + \ell_p + \ell'_p - fv_p$, we are looking at the quantity $v_c + \ell_p + \ell'_p - fv_p - \hat{v}_c - ((\ell_p + \ell'_p) - bv_c) = v_c - fv_p - \hat{v}_c + bv_c$. The number of vetoes for c after bribery is $v_c - \hat{v}_c + bv_c$, and this number is $\geq fv_p$, since p is a winner. It follows that the number of padding edges incident with c in E' is nonnegative.
- The number of edges in E' incident with y is

$$\begin{aligned}\sum_{c \in C - \{p\}} ((\ell_p + \ell'_p) - bv_c) &= \sum_{c \in C - \{p\}} (\ell_p + \ell'_p) - \sum_{c \in C - \{p\}} bv_c \\ &= \|C - \{p\}\|(\ell_p + \ell'_p) - 2(\ell_p + \ell'_p) \\ &= b(y).\end{aligned}$$

  This is nonnegative, since we have at least three candidates.
- The number of edges in E' incident with x is the number of padding edges in E', which is nonnegative by the argument for $c \in C - \{p\}$ above. This quantity is

$$\sum_{c \in C - \{p\}} (b(c) - \hat{v}_c - ((\ell_p + \ell'_p) - bv_c)) =$$
$$\sum_{c \in C - \{p\}} b(c) - \sum_{c \in C - \{p\}} \hat{v}_c - \sum_{c \in C - \{p\}} ((\ell_p + \ell'_p) - bv_c).$$

Since

$$\sum_{c \in C - \{p\}} \hat{v}_c = \ell_p + 2\ell'_p$$

and

$$\sum_{c \in C - \{p\}} ((\ell_p + \ell'_p) - bv_c) = b(y),$$

this is exactly $b(x)$.

For the converse, suppose G has a perfect matching of weight k. Let $\widehat{V}$ be the collection of voters corresponding to the voter-edges in the matching. We will show that we can make p a winner by bribing the voters in $\widehat{V}$. Note that the cost of this bribery is k. For $c \in C - \{p\}$, let bv_c be the number of bribe-edges incident with c that are not in the perfect matching. We will show that

1. If the bribery gives bv_c vetoes to each candidate $c \in C - \{p\}$ and no vetoes to p, then p is a winner, and
2. there are $\ell_p + \ell'_p$ votes that give bv_c vetoes to each candidate $c \in C - \{p\}$ and no vetoes to p.

The second item follows immediately from the fact that $\sum_{c \in C - \{p\}} bv_c = 2(\ell_p + \ell'_p)$ (since there are exactly $2(\ell_p + \ell'_p)$ edges incident with y that are not in the matching), and for each $c \in C - \{p\}, bv_c \leq \ell_p + \ell'_p$ (since there are $\ell_p + \ell'_p$ bribe-edges incident with c in G).

For the first item, we need to show that for each candidate $c \in C - \{p\}$, $v_c - \hat{v}_c + bv_c$ (the number of vetoes for c after bribery) is at least fv_p.

From the matching, we have that for each candidate $c \in C - \{p\}$, $\hat{v}_c$ (the number of voter-edges incident with c in the matching) $+(\ell_p + \ell'_p - bv_c)$ (the number of bribe-edges incident with c in the matching) $\leq b(c) = v_c + \ell_p + \ell'_p - fv_p$. This implies that

$$\hat{v}_c - bv_c \leq v_c - fv_p,$$

which immediately implies that

$$fv_p \leq v_c - \hat{v}_c + bv_c,$$

and so p is a winner. □

5 3-Veto-$Bribery is NP-complete

It should come as no surprise that 3-Veto-$Bribery is NP-complete. This was claimed in Bredereck and Talmon [5], but with an incorrect proof (a simple modification of their incorrect NP-hardness proof for 2-Veto-$Bribery).

Theorem 5. *3-Veto-$Bribery is* NP*-complete.*

Proof. We give a straightforward reduction from Restricted Exact Cover by 3-Sets to our bribery problem. The well-known NP-complete problem of Exact Cover by 3-Sets (X3C) asks, given a finite collection of 3-element subsets of some finite set B, whether the collection has an exact cover, i.e., whether there exists a subcollection such that each element of B occurs exactly once [17]. We reduce from Restricted Exact Cover by 3-Sets (RX3C), which is the restriction of X3C where each element of B occurs in exactly three subsets. This restricted problem is still NP-complete [16].

Name: Restricted Exact Cover by 3-Sets (RX3C)

Given: Given a set $B = \{b_1, \ldots, b_{3k}\}$ and a collection $\mathcal{S} = \{S_1, \ldots, S_n\}$ of 3-element subsets of B where each $b \in B$ occurs in exactly three subsets S_i (so $n = 3k$).

Question: Does there exist a subcollection $\mathcal{S}'$ of $\mathcal{S}$ that forms an exact cover of B?

Given an instance of RX3C, $B = \{b_1, \ldots, b_{3k}\}$ and $\mathcal{S} = \{S_1, \ldots, S_n\}$, we construct an instance of 3-Veto-$Bribery in the following way.

The set of candidates consists of B, p, two padding candidates p_1 and p_2, and $3k$ dummy candidates $d_1, \ldots, d_{3k}$. The preferred candidate is p and the budget is k. The voters are defined as follows.

- For each set $S \in \mathcal{S}$ with $S = \{x, y, z\}$ we have one voter that vetoes x, y, and z, with price 1.
- We have two voters vetoing p_1, p_2, and p, each with price $k + 1$.
- We have k voters, each with price $k + 1$, with votes that ensure each of the d_i candidates receive one veto.

Note that each dummy candidate has one veto, p and the two padding candidates have two vetoes, and each candidate in B has three vetoes.

If p can be made a winner by bribing a subcollection of voters with total price at most k we will show that there is an exact cover $\mathcal{S}'$ of $\mathcal{S}$. Only the voters with votes corresponding to sets in $\mathcal{S}$ can be bribed, since all other voters have prices greater than the budget. This means that p will have at least two vetoes in any bribed election, and a successful bribery would need each of the $3k$ dummy candidates d_i to gain a veto and each b_i candidate can lose at most one veto. Therefore voters corresponding to sets in $\mathcal{S}$ will correspond to an exact cover.

For the converse, suppose there exists an exact cover $\mathcal{S}'$ of $\mathcal{S}$. Since each $b \in B$ occurs exactly three times in $\mathcal{S}$ we know $\|\mathcal{S}'\| = k$. We can construct a successful bribery by bribing the k voters corresponding to $\mathcal{S}'$ to veto the $3k$ dummy candidates d_i. Since $\mathcal{S}'$ is an exact cover, each b_i candidate will lose exactly one veto, and so p is a winner. □

6 Pushing the Boundary

As we've seen in the previous section, weighted matching turned out to be a useful tool to solve some open control and bribery problems. These problems inherit their complicated algorithms from weighted matching. In this section we discuss how far we can push this technique; can we provide polynomial-time algorithms for more general problems?

It turns out that we can. For the sake of such an example we will show that even after adding prices to the registered and unregistered voters in control by replacing voters for 2-Approval, this problem will remain in P.[4] Priced control was introduced by Miasko and Faliszewski [21] for control by adding and deleting voters and candidates. In priced control, each control action has a price and the question is whether it is possible to obtain the desired result by control within a budget. We can extend this notion to control by replacing voters, by having each registered voter $v \in V$ have a price $\pi(v)$, each unregistered voter $w \in W$ have price $\pi(w)$, and letting k be the budget.

We will now explain how to modify the proof of Theorem 3 using some of the ideas from the proof of Theorem 4 to also work for priced control by replacing voters for 2-Approval. Unlike in the unpriced case, we do not know what fs_p will be. And so, similarly as in the proof of Theorem 4, we will brute-force over all (polynomially many) possible values of fs_p. For each value of fs_p, $0 \leq fs_p \leq \|V\|$, we look at the graph defined in the proof of Theorem 3. The only difference will be the weights. All we have to do is to set the weight of each voter-edge that corresponds to a voter $v \in V$ to $\pi(v)$ instead of 1 and each voter-edge that corresponds to a voter $w \in W$ to $-\pi(w)$ instead of 0. The same argument as in the proof of Theorem 3 shows that there is a successful control action within budget k if and only if G has a perfect matching of weight $\pi(V) - k$.

An immediate corollary of the above is that this implies that 2-Approval-$Bribery is in P (as Bredereck et al. [4] state). We can solve 2-Approval-$Bribery using the result just stated for priced control by replacing voters: The registered voters (including price) will be the n voters from the 2-Approval-$Bribery instance that we want to solve, the budget will be same, and the unregistered voters will consist of "all possible votes" that we can bribe to, each with price 0, i.e., for each pair of candidates a, b, the collection of unregistered voters contains n voters approving a and b, each with price 0.

7 Conclusion

We showed that by using weighted matching we were able to solve the sole open problem from the comprehensive study of control by Erdélyi et al. [8]. We also used weighted matching to solve the remaining standard bribery cases for k-Veto. Overall this settles the complexity for k-Approval and k-Veto for standard control and bribery cases. Unlike unweighted matching, weighted matching is not often used to solve voting problems (though it was, e.g., recently used in a multiwinner election setting [6] and for bipartite graphs to compute the distance between elections [12]), but as our results show, it is a powerful technique whose potential deserves to be further explored.

Acknowledgements

This work was supported in part by NSF-DUE-1819546. We thank Robert Bredereck for helpful discussions and the anonymous reviewers for their comments and suggestions.

[4] Though it may seem from our results that control problems for 2-Approval will all be in P this is not the case. Control by adding and control by deleting candidates are each NP-complete [18, 19], and weighted control by deleting voters is NP-complete [10].

References

[1] J. Bartholdi, III, C. Tovey, and M. Trick, 'How hard is it to control an election?', *Mathematical and Computer Modelling*, **16**(8/9), 27–40, (1992).

[2] C. Berge, *Graphs and Hypergraphs*, North-Holland Publishing Company, 1973.

[3] F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. Procaccia, *Handbook of Computational Social Choice*, Cambridge University Press, 2016.

[4] R. Bredereck, J. Goldsmith, and G. Woeginger, 'Working group: (control and) bribery in k-approval voting—open problems', *Dagsthul Reports*, **5**, 24, (2016).

[5] R. Bredereck and N. Talmon, 'NP-hardness of two edge cover generalizations with applications to control and bribery for approval voting', *Information Processing Letters*, 147–152, (2016).

[6] L. Celis, L. Huang, and N. Vishnoi, 'Multiwinner voting with fairness constraints', in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 144–151, (July 2018).

[7] J. Edmonds and E. Johnson, 'Matching: a well-solved class of integer linear programs', in *Combinatorial structures and their applications (Gordon and Breach)*, pp. 89–92, (1970).

[8] G. Erdélyi, M. Neveling, C. Reger, J. Rothe, Y. Yang, and R. Zorn, 'Towards completing the puzzle: Complexity of control by replacing, adding, and deleting candidates or voters', *Autonomous Agents and Multi-Agent Systems*, **35**(41), 1–48, (2021).

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

[10] P. Faliszewski, E. Hemaspaandra, and L. Hemaspaandra, 'Weighted electoral control', *Journal of Artificial Intelligence Research*, **52**, 507–542, (2015).

[11] P. Faliszewski and J. Rothe, 'Control and bribery in voting', in *Handbook of Computational Social Choice*, 146–168, Cambridge University Press, (2016).

[12] P. Faliszewski, P. Skowron, A. Slinko, S. Szufa, and N. Talmon, 'How similar are two elections?', in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pp. 1909–1916, (January 2019).

[13] Z. Fitzsimmons and E. Hemaspaandra, 'Insight into voting problem complexity using randomized classes', in *Proceedings of the 31st International Joint Conference on Artificial Intelligence*, pp. 293–299, (July 2022).

[14] Z. Fitzsimmons and E. Hemaspaandra, 'Using weighted matching to solve 2-Approval/Veto control and bribery', Tech. Rep. arXiv:2305.16889 [cs.GT], arXiv.org, (May 2023).

[15] A. Gerards, 'Matching', in *Handbooks in OR and MS Vol. 7*, eds., M. Ball, T. Magnanti, C. Monma, and G. Nemhauser, chapter 3, 135–224, Cambridge University Press, (1995).

[16] T. Gonzalez, 'Clustering to minimize the maximum intercluster distance', *Theoretical Computer Science*, **38**, 293–306, (1985).

[17] R. Karp, 'Reducibility among combinatorial problems', in *Proc. of Symposium on Complexity of Computer Computations*, pp. 85–103, (1972).

[18] A. Lin, 'The complexity of manipulating k-approval elections', in *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence*, pp. 212–218, (January 2011).

[19] A. Lin, *Solving Hard Problems in Election Systems*, Ph.D. dissertation, Rochester Institute of Technology, Rochester, NY, 2012.

[20] A. Loreggia, N. Narodytska, F. Rossi, K. Venable, and T. Walsh, 'Controlling elections by replacing candidates or votes', in *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, pp. 1737–1738, (May 2015).

[21] T. Miasko and P. Faliszewski, 'The complexity of priced control in elections', *Annals of Mathematics and Artificial Intelligence*, 1–26, (2014).

[22] K. Mulmuley, U. Vazirani, and V. Vazirani, 'Matching is as easy as matrix inversion', *Combinatorica*, **7**(1), 105–113, (1987).

[23] C. Papadimitriou and M. Yannakakis, 'The complexity of restricted spanning tree problems', *Journal of the ACM*, **29**(2), 285–309, (1982).

[24] W. Tutte, 'A short proof of the factor theorem for finite graphs', *Canadian Journal of Mathematics*, **6**, 347–352, (1954).