Insight into Voting Problem Complexity Using Randomized Classes

Zack Fitzsimmons[1] and **Edith Hemaspaandra**[2]
[1]College of the Holy Cross
[2]Rochester Institute of Technology
zfitzsim@holycross.edu, eh@cs.rit.edu

Abstract

The first step in classifying the complexity of an NP problem is typically showing the problem in P or NP-complete. This has been a successful first step for many problems, including voting problems. However, in this paper we show that this may not always be the best first step. We consider the problem of constructive control by replacing voters (CCRV) introduced by Loreggia et al. [2015] for the scoring rule First-Last, which is defined by $\langle 1, 0, \ldots, 0, -1 \rangle$. We show that this problem is equivalent to Exact Perfect Bipartite Matching, and so CCRV for First-Last can be determined in random polynomial time. So on the one hand, if CCRV for First-Last is NP-complete then RP = NP, which is extremely unlikely. On the other hand, showing that CCRV for First-Last is in P would also show that Exact Perfect Bipartite Matching is in P, which would solve a well-studied 40-year-old open problem.

Considering RP as an option for classifying problems can also help classify problems that until now had escaped classification. For example, the sole open problem in the comprehensive table from Erdélyi et al. [2021] is CCRV for 2-Approval. We show that this problem is in RP, and thus easy since it is widely assumed that P = RP.

1 Introduction

Elections are an important tool used to aggregate the preferences of several agents (voters) over a set of choices (candidates) with applications in areas such as political domains and multiagent systems in artificial intelligence settings. We consider computational problems relating to elections. Specifically the computational complexity of electoral control.

Electoral control models the actions of an agent with control over an election, referred to as the chair, who modifies this structure to ensure a preferred outcome, e.g., by adding voters to the election to ensure a preferred candidate wins. Types of electoral control model many different types of real-world manipulative actions on elections such as get-out-the-vote drives. The study of electoral control was introduced by Bartholdi, Tovey, and Trick [1992] who studied the computational complexity of different types of control, including control by adding voters, for several voting rules. This important initial work led to lots of subsequent work on control (see, e.g., Faliszewski and Rothe [2016]).

A natural model of control introduced by Loreggia et al. [2015] is constructive control by replacing voters (CCRV) in which the election chair (for example to avoid detection) replaces a set of voters from the election with the same number of unregistered voters in order to ensure a preferred candidate wins. This model has been thoroughly studied and recent work by Erdélyi et al. [2021] completes many open cases and includes a comprehensive table of known results. The sole open case is CCRV for 2-Approval elections.

When studying an electoral control problem for a given voting rule, the first step is typically to determine if the action is in P or NP-complete. If the problem is NP-complete, the next steps could involve empirical approaches, parameterized complexity, or approximation (see, e.g., Rothe and Schend [2013] and Dorn and Schlotter [2017]). However, some problems in NP do not seem to easily be classified as being in P or NP-complete. An example of such a problem is Exact Perfect Matching (in which we ask if there exists a perfect matching with exactly a certain number of red edges) introduced by Papadimitriou and Yannakakis [1982] who conjectured it to be NP-complete. Mulmuley, Vazirani, and Vazirani [1987] later gave a randomized polynomial-time algorithm thus showing the problem easy.

We consider control by adding voters (CCAV) and control by replacing voters (CCRV) as well as their exact variants (CCAV! and CCRV!) for the scoring rules First-Last (defined by $\langle 1, 0, \ldots, 0, -1 \rangle$) and 2-Approval, and link their complexity to the complexity of Exact Perfect Matching. Our results are summarized in Table 1. Equivalence in Table 1 means polynomial-time disjunctive truth-table (dtt) equivalence (a more flexible notion than many-one equivalence; see the Preliminaries) though many of the individual reductions are proven with less flexible (typically logspace many-one) reductions. Since RP is closed under dtt reductions, and Exact Perfect Bipartite Matching (EPBM) and Exact Perfect Matching are in RP, the problems polynomial-time dtt equivalent to these problems are also in RP. If one of the equivalent problems is NP-complete, then $\mathrm{RP} = \mathrm{NP}$, which is very unlikely. On the other hand, if one of the equivalent problems is in

P, then so is EPBM, which would solve a 40-year-old open problem.[1]

Our results establish even better upper bounds than RP, since EPBM and Exact Perfect Matching are not only in RP, but even in RNC [Mulmuley *et al.*, 1987], which is the class of problems with efficient randomized parallel algorithms.

Our work is the first to show equivalence between voting problems and Exact Perfect [Bipartite] Matching. In fact the only work we found that does something related in voting is Giorgos [2019], which reduces a multiwinner election problem to Exact Perfect Matching (but does not show equivalence). Our approach of showing equivalence to Exact Perfect [Bipartite] Matching may be useful for other voting problems that have thus-far defied classification as being in P or NP-complete.

2 Preliminaries

An election (C, V) is a pair comprised of a set of candidates C and a set of voters V where each voter $v \in V$ has a total-order vote over the set candidates.

A voting rule $\mathcal{E}$ is a mapping from an election to a set of candidates referred to as the winners. Our results are for two natural scoring rules, First-Last and 2-Approval. A scoring rule is defined by a family of scoring vectors of the form $\langle \alpha_1, \alpha_2, \ldots, \alpha_m \rangle$ (where m is the number of candidates) with $\alpha_i \geq \alpha_{i+1}$ where a candidate ranked ith by a voter receives score α_i from that voter. The rule First-Last is described by the general scoring vector $\langle 1, 0, \ldots, 0, -1 \rangle$ and 2-Approval is described by the general scoring vector $\langle 1, 1, 0, \ldots, 0 \rangle$.

2.1 Election Problems

We examine the complexity of two types of electoral control, namely, control by adding voters and control by replacing voters. Control by adding voters was introduced by Bartholdi, Tovey, and Trick [1992], and control by replacing voters was introduced by Loreggia et al. [2015].

Name: $\mathcal{E}$-Constructive Control by Adding Voters (CCAV)
Given: An election (C, V), a set of unregistered voters W, an integer $k \geq 0$, and a preferred candidate p.
Question: Does there exist a set $W' \subseteq W$ such that $\|W'\| \leq k$ and p is an $\mathcal{E}$-winner of the election $(C, V \cup W')$?

Name: $\mathcal{E}$-Constructive Control by Replacing Voters (CCRV)
Given: An election (C, V), a set of unregistered voters W, an integer $k \geq 0$, and a preferred candidate p.
Question: Do there exist sets $V' \subseteq V$ and $W' \subseteq W$, such that $\|V'\| = \|W'\| \leq k$ and p is an $\mathcal{E}$-winner of the election $(C, (V - V') \cup W')$?

We also consider the *exact* versions of these problems (introduced by Erdélyi [2021]) where we ask if it is possible to add or replace exactly k voters. We refer to these problems as CCAV! and CCRV!, respectively.

[1] One might wonder why we (and others) do not prove such problems complete for a class such as RP. The answer to that is that semantically defined classes such as RP may not have complete problems. In particular, RP does not robustly (i.e., in every relativized world) possess many-one complete sets [Sipser, 1982] or even Turing-complete sets [Hemaspaandra *et al.*, 1993]. Of course if $\mathrm{RP} = \mathrm{P}$, then RP does have complete sets.

2.2 Exact Perfect Matching

This paper explores the connection between the abovementioned election problems with the following graph problems introduced by Papadimitriou and Yannakakis [1982].

Name: Exact Perfect [Bipartite] Matching
Given: A [bipartite] graph $G = (V, E)$, a set $R \subseteq E$ of red edges, and an integer $k \geq 0$.
Question: Does there exist a perfect matching of G with exactly k red edges, i.e., a set of edges $E' \subseteq E$, exactly k of which are red, such that each $v \in V$ is incident with exactly one edge in E'?

In proving our results we use several different intermediate problems and define these when they are first used.

2.3 Computational Complexity

We assume that the reader is familiar with the classes P and NP, and what it means for a problem to be NP-complete. NP-completeness is typically shown using polynomial-time many-one reductions. However, there are many other types of reductions. Many of our results will use logspace many-one reductions (where the reduction is even computable in logspace) as well as disjunctive truth-table reductions. A disjunctive truth-table (dtt) reduction is a generalization of the many-one reduction. A dtt reduction from X to Y outputs a list of strings such that $x \in X$ if and only if at least one of the strings in the list is in Y. Two languages X and Y are equivalent with respect to a given type of reduction if X reduces to Y and Y reduces to X, and the type of equivalence (e.g., polynomial-time dtt equivalence) is determined by the most flexible reduction used in establishing the equivalence.

RP (sometimes called R) is the class of languages L for which there exists a probabilistic polynomial-time algorithm A such that for all $x \in L$, A accepts with probability at least $1/2$, and for all $x \notin L$, A always rejects [Gill, 1977]. Note that A can have false negatives, but not false positives, and by iterating we can make the probability of a false negative exponentially small.

RNC is a subset of RP, and is defined as the class of languages that can be decided by a randomized algorithm in polylogarithmic time on a parallel computer with a polynomial number of processors (see, e.g., Papadimitriou [1994]). Since RNC is not known to contain P, it may not be closed under polynomial-time reductions, but RNC is closed under logspace dtt reductions.

We note in passing that the "P" upper bounds listed in Table 1 could be replaced by $\mathrm{P} \cap \mathrm{RNC}$, since 2-Approval CCAV! logspace many-one reduces to 2-Approval CCRV by padding (after some preprocessing), CCAV always logspace dtt reduces to CCAV!, and we show that CCRV for 2-Approval and First-Last are in RNC.

For a more detailed description of the concepts introduced in this section, see, e.g., Papadimitriou [1994].

3 First-Last Elections

In this section, we will show that CCRV, CCAV!, and CCRV! for First-Last are polynomial-time dtt equivalent to EPBM. These are the first voting problems equivalent to EPBM. This implies that we are very unlikely to be able to classify these

	First-Last	2-Approval
CCAV	P [Hemaspaandra *et al.*, 2014]	P [Lin, 2011]
CCAV!	equiv. to Exact Perfect Bipartite Matching (Section 3.1)	P (Theorem 10)
CCRV	equiv. to Exact Perfect Bipartite Matching (Section 3.2)	reduces to Exact Perfect Matching (Corollary 11)
CCRV!	equiv. to Exact Perfect Bipartite Matching (Section 3.2)	equiv. to Exact Perfect Matching (Section 4)

Table 1: Summary of our results

control problems as being in P or being NP-complete: If First-Last CCAV!/CCRV!/CCRV is NP-complete, then $\mathrm{RP} = \mathrm{NP}$, which is generally believed to be almost as unlikely as $\mathrm{P} = \mathrm{NP}$. And if we were to prove CCAV!/CCRV!/CCRV to be in P, then we would also have shown that EPBM is in P. The proof consists of a large number of reductions, some of which are complicated. To make the proof easier to follow, we will first prove the CCAV! case, which shows the main ideas, and then show how to modify this for CCRV! and CCRV.

3.1 First-Last-CCAV! Is Equivalent to EPBM

The equivalence of First-Last-CCAV! to EPBM follows from the following cycle of reductions:

1. Exact Cycle Sum logspace many-one reduces to First-Last-CCAV! (Theorem 1).
2. First-Last-CCAV! logspace many-one reduces to Exact Perfect Bipartite b-Matching (Theorem 3).
3. Exact Perfect Bipartite b-Matching logspace many-one reduces to EPBM (Theorem 5).
4. EPBM polynomial-time dtt reduces to Exact Cycle Sum [Papadimitriou and Yannakakis, 1982, Proposition 1].[2]

The above also implies that First-Last-CCAV! is in RNC, since items 2 and 3 imply that First-Last-CCAV! logspace many-one reduces to EPBM, RNC is closed under logspace dtt reductions (and so certainly under logspace many-one reductions), and EPBM is in RNC [Mulmuley *et al.*, 1987]. Exact Cycle Sum is defined as follows [Papadimitriou and Yannakakis, 1982].

Name: Exact Cycle Sum
Given: Digraph G and integer $k \geq 0$.
Question: Is there a set of (vertex) disjoint cycles of total length exactly k?

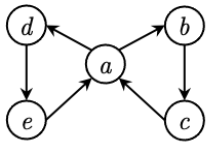
Exact Perfect Bipartite b-Matching is defined as follows.

Name: Exact Perfect Bipartite b-Matching
Given: A bipartite multigraph $G = (V, E)$, a capacity function $b : V \to \mathbb{N}$, a set $R \subseteq E$ of red edges, and an integer $k \geq 0$.
Question: Does there exist a perfect b-matching of G with exactly k red edges, i.e., a set of edges $E' \subseteq E$, exactly k of which are red, such that each $v \in V$ is incident with exactly $b(v)$ edges in E'?

Theorem 1 *Exact Cycle Sum logspace many-one reduces to First-Last-CCAV!.*

Proof. We will first show that the obvious attempt at a reduction fails. Let G be a digraph. Let the candidates be $V(G)$ plus preferred candidate p. There are no registered voters. For each arc (a, b) in G, we have an unregistered voter $b > \cdots > a$ (i.e., ranking b first and a last) and we ask if we can add exactly k voters such that p is a winner (note that in that case all candidates are tied at 0). It is easy to see that a set of disjoint cycles of total length exactly k corresponds to a set of k added voters such that p is a winner. However, the following example shows that the converse does not hold.

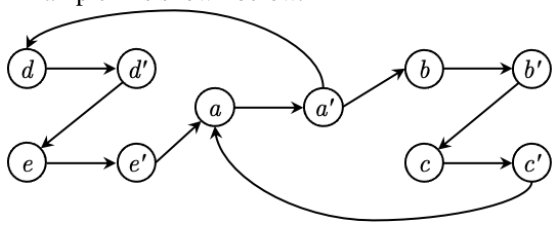
Example 2 *Consider the digraph consisting of arcs (a, b), (b, c), (c, a), (a, d), (d, e), and (e, a) (that is, two cycles of length 3, intersecting in a) and the corresponding set of voters.*



$b > \cdots > a$
$c > \cdots > b$
$a > \cdots > c$
$d > \cdots > a$
$e > \cdots > d$
$a > \cdots > e$

There is no set of disjoint cycles of total length exactly 6 in the given graph. However, if we add all six corresponding voters, p is a winner.

Note that the reduction attempt above *will* successfully reduce the version of Exact Cycle Sum in which the cycles are edge-disjoint rather than (vertex-)disjoint to First-Last-CCAV!. And it is easy to see this reduction is computable in logspace, since each voter corresponds directly to an arc. It remains to show that we can reduce Exact Cycle Sum to the edge-disjoint version of Exact Cycle Sum (in logspace). This is not hard. Given a digraph G, we create a new digraph G' as follows. For every vertex v in G, we have an arc (v, v') in G'. For every arc (v, w) in G, we have an arc (v', w) in G'. Set the sum to $2k$. An example of this construction on the graph from Example 2 is shown below.



Every cycle that goes through v must go through (v, v') and so a set of disjoint cycles in G corresponds to a set of edge-disjoint cycles in G' of twice the length. ❑

[2] Papadimitriou and Yannakakis [1982] only consider "polynomial equivalence," though it is clear from inspection that some of the reductions in their paper are in fact logspace many-one reductions. This particular reduction however computes a perfect bipartite matching (which can be done in polynomial time, but it is not known if this can be done in logarithmic space; we mention that recent work shows this problem is in quasi-NC [Fenner *et al.*, 2021]). Careful inspection also shows that the reduction from Papadimitriou and Yannakakis [1982] is a dtt but not a many-one reduction.

Theorem 3 *First-Last-CCAV! logspace many-one reduces to Exact Perfect Bipartite b-Matching.*

Proof. Let p be the preferred candidate and let k be the number of voters to add. If there are at least k unregistered voters with p first, we will add only voters that have p first. Then the final score of p will be the score of p from the registered voters plus k and we can easily check if we can add k such voters in such a way that the score of each candidate $a \neq p$ is at most the final score of p.

If there are $\ell < k$ unregistered voters with p first, we add all of those (to the registered voters) and subtract ℓ from k. So, all remaining unregistered voters do not have p first.

Note that if we have not yet determined if control is possible, we are left with an instance in which no unregistered voter has p first (note that in the process, k may have been updated). If $\ell' < k$ voters do not have p last, add all of those and check if you can add $k - \ell'$ voters that have p last such that p is still a winner. If there are at least k voters that do not have p last, we will never add a voter that has p last, so delete all unregistered voters that have p last.

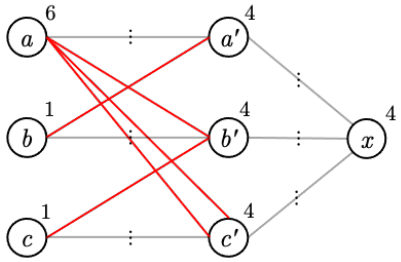
Note that after this preprocessing, we have either determined whether control is possible, or we are left with an instance where all unregistered voters give 0 points to p.

Let s_c be the score of c from the registered voters. We create the following graph (see Example 4 for an example of the construction). The vertices are a, a' for every candidate $a \neq p$ and a special vertex x. The graph will be bipartite, with the nonprimed vertices (including x) in one part and the primed vertices in the other.

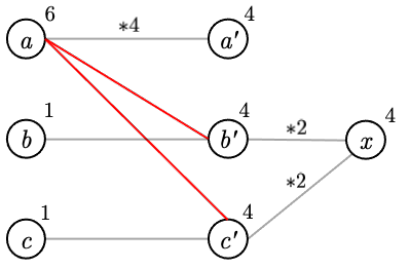
For every unregistered voter voting $b > \cdots > a$, we have a red edge (a, b'). Note that this will give a multigraph, since we can have multiple voters with the same vote. We also have an "infinite" number of nonred edges (a, a') and an "infinite" number of nonred edges (x, a'), where infinite means as many as we could ever need.

Let M be a constant that is large enough so that all b-values (to be specified below) are positive. We set the b-values as follows: $b(a') = M$; $b(a) = M + s_a - s_p$; $b(x)$ is set so that the sum of the unprimed b-values = the sum of the primed b-values, i.e., $b(x) = \sum_{a \neq p}(s_p - s_a)$. If this is negative, then there is no solution.

Example 4 *Consider an instance of First-Last-CCAV!, after the preprocessing described above has been performed, with preferred candidate p, $k = 2$, registered voters resulting in the following scores: $s_a = 3$, $s_b = -2$, $s_c = -2$, and $s_p = 1$, and five unregistered voters: $b > \cdots > a$, $c > \cdots > a$, $c > \cdots > a$, $b > \cdots > c$, and $a > \cdots > b$. Below is the corresponding bipartite graph as described in the construction.*



p wins after adding the voter voting $b > \cdots > a$ and one of the voters voting $c > \cdots > a$. This corresponds to the following perfect matching with exactly two red edges.



We will show that we can add k unregistered voters such that p becomes a winner if and only if there exists a perfect matching that contains exactly k red edges.

If there is a set of k unregistered voters that we can add so that p becomes a winner, then add the edges corresponding to the added voters to the matching. These are k red edges. We need to show that we can extend this to a perfect matching by adding just nonred edges.

For $a \neq p$, let F_a be the set of added voters with a first, and let L_a be the set of added voters with a last. Since p is a winner, we know that $s_a + \|F_a\| - \|L_a\| \leq s_p$ $(*)$.

The edges corresponding to L_a are incident with a, and the edges corresponding to F_a are incident with a'. $b(a) = M + s_a - s_p$, and so we need $M + s_a - s_p - \|L_a\|$ nonred edges incident with a to get a perfect matching. All nonred edges incident with a go to a', and there are as many of them as we need. The only thing we need to ensure is that a' can handle that many nonred edges. $b(a') = M$ and a' is already incident with $\|F_a\|$ red edges. So we need that $M + s_a - s_p - \|L_a\| \leq M - \|F_a\|$. This follows from $(*)$. We then make sure that a' is incident with exactly $b(a')$ edges by adding as many (x, a') edges as needed.

For the converse, suppose there is a perfect bipartite b-matching with exactly k red edges. For each red edge (a, b') add the corresponding voter (who votes $b > \cdots > a$). We claim that p is a winner. Consider candidate $a \neq p$. Let L_a be the set of red edges incident with a and let F_a be the set of red edges incident with a'. We know that there are $M + s_a - s_p - \|L_a\|$ nonred edges incident with a and that there are $M - \|F_a\|$ nonred edges incident with a'. Since all nonred edges incident with a go to a', it follows that $M + s_a - s_p - \|L_a\| \leq M - \|F_a\|$, which implies that $s_a - s_p + \|F_a\| - \|L_a\| \leq 0$, and so the score of a after addition, $s_a + \|F_a\| - \|L_a\|$, is at most s_p. ❑

Theorem 5 *Exact Perfect [Bipartite] b-Matching logspace many-one reduces to Exact Perfect [Bipartite] Matching.*

The proof of the above theorem generalizes the reduction from Perfect b-Matching to Perfect Matching (see, e.g., Berge [1973]) and can be found in the full version [Fitzsimmons and Hemaspaandra, 2022].

3.2 First-Last CCRV! and CCRV are Equivalent to EPBM

We will now show that CCRV! and CCRV for First-Last are equivalent to EPBM. This follows from the following cycle

of reductions.

1. First-Last-CCAV! logspace many-one reduces to First-Last-CCRV (Theorem 7).
2. First-Last-CCRV logspace dtt reduces to First-Last-CCRV! (Observation 6).
3. First-Last-CCRV! logspace many-one reduces to EPBM (Theorems 8 and 9).
4. EPBM polynomial time dtt reduces to First-Last-CCAV! (previous section).

The above also implies that First-Last-CCRV and First-Last-CCRV! are each in RNC.

We start by noting that control by replacing at most k voters is possible if and only if for some ℓ, $0 \leq \ell \leq k$, control by replacing exactly ℓ voters is possible.

Observation 6 *For any voting rule X, X-CCRV logspace dtt reduces to X-CCRV!.*

Next, we will show that First-Last-CCAV! reduces to First-Last-CCRV.

Theorem 7 *First-Last-CCAV! logspace many-one reduces to First-Last-CCRV (and to First-Last-CCRV!).*

Proof. Consider an instance of First-Last-CCAV! with registered voters V, unregistered voters W, preferred candidate p, and limit k. As in the proof of Theorem 3 it suffices to consider First-Last-CCAV! where p receives 0 points from each unregistered voter.

Let s_p denote the score of p from the registered voters. Note that in a First-Last election the sum of the scores of all candidates is 0. And so if s_p is negative, p cannot be made a winner by exact adding.

So assume $s_p \geq 0$. We now construct an instance of First-Last-CCRV as follows. Let the candidate set consist of C with $2k$ additional candidates: $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$. Update the preferences for each voter so that these candidates receive 0 points from each registered and unregistered voter. Except for adding these new candidates to the votes, the set of unregistered voters remains the same. However, for each i, $1 \leq i \leq k$, add $s_p + 1$ voters voting $a_i > \cdots > b_i$ to the set of registered voters. Now each a_i candidate has score $s_p + 1$, each b_i candidate has score $-s_p - 1$, and the scores of the remaining candidates are unchanged.

If control is possible by adding exactly k voters such that p wins, then control by replacing voters is possible by replacing k voters, with the ith voter voting $a_i > \cdots > b_i$ for $1 \leq i \leq k$, with those same k unregistered voters. This decreases the score of each a_i candidate by 1 so that they tie with p, and it is easy to see that p wins.

For the converse, suppose that there is a way to replace at most k voters such that p wins. Since the score of each of the k a_i candidates must decrease by 1, and each unregistered voter gives 0 points to a_i, at least one of the voters voting $a_i > \cdots > b_i$ must be replaced for each i. It follows that we replace exactly one voter voting $a_i > \cdots > b_i$ for each i, and it is straightforward to see that the k unregistered voters replacing the a_i voters correspond to k voters that can be added to V so that p wins by exact control by adding voters.

The same reduction reduces to CCRV!. ❑

And finally, we will show that First-Last-CCRV! reduces to EPBM. Note that we can view CCRV! as CCAV! where we have two sets of unregistered voters. This suggests reducing to the following variation of Exact Perfect Bipartite b-Matching.

Name: Exact Red-Blue Perfect Bipartite b-Matching
Given: A bipartite multigraph $G = (V, E)$, a capacity function $b : V \to \mathbb{N}$, disjoint sets $R \subseteq E$ of red edges and $B \subseteq E$ of blue edges, an integer $k \geq 0$, and an integer $\ell \geq 0$.
Question: Does there exist a perfect b-matching of G with exactly k red edges and exactly ℓ blue edges, i.e., a set of edges $E' \subseteq E$, exactly k of which are red and exactly ℓ of which are blue, such that each $v \in V$ is incident with exactly $b(v)$ edges in E'?

Theorem 8 *First-Last-CCRV! logspace many-one reduces to Exact Red-Blue Perfect Bipartite b-Matching.*

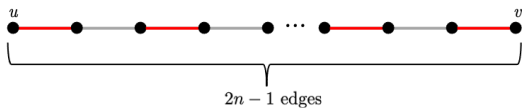
The proof of the above theorem can be found in the full version.

Theorem 9 *Exact Red-Blue Perfect [Bipartite] b-Matching logspace many-one reduces to Exact Perfect [Bipartite] Matching.*

Proof. The construction from the proof of Theorem 5 (when we color edges in the constructed graph with the color of their corresponding edges in the multigraph) reduces Exact Red-Blue Perfect [Bipartite] b-Matching to Exact Red-Blue Perfect [Bipartite] Matching.

It remains to reduce Exact Red-Blue Perfect [Bipartite] Matching to Exact Perfect [Bipartite] Matching.

Let G be the graph with red and blue edges, and let k and ℓ be the exact numbers of red and blue edges we want in the perfect matching. Without loss of generality, assume that $k, \ell < n$, where n is the number of vertices (note that a perfect matching contains $n/2$ edges). Adapting approaches from Papadimitriou and Yannakakis [1982, Proposition 1], we replace each blue edge (u, v) by a path of length $2n-1$ colored red-nonred alternately:



Call this graph G'. Note that G' is bipartite if G is bipartite and that G' uses only the color red. We claim that G has a perfect matching with k red and ℓ blue edges if and only if G' has a perfect matching with $\ell n + k$ red edges. Consider a perfect matching of G with k red and ℓ blue edges. We will modify this to obtain a perfect matching for G'. For each blue edge in G consider the length $2n - 1$ path in G' that corresponds to this edge. If the blue edge is in the matching of G, replace this edge by the n red edges on the path. If the blue edge is not in the matching, put the $n - 1$ nonred edges on the path in the matching. This gives a perfect matching of G' with the required number of red edges. For the converse, consider a perfect matching of G' with $\ell n + k$ red edges. Note that every path corresponding to a blue edge in G contributes n or 0 red edges to the perfect matching of G'. Also note

that the matching of G' contains fewer than n nonpath edges. It follows that exactly ℓ of the paths corresponding to a blue edge contribute n red edges to the matching. These are exactly the blue edges in the perfect matching for G. And the remaining edges in the matching of G' (k of which are red) are the remaining edges in the perfect matching of G. ❑

4 2-Approval Elections

In the previous section we found that control by replacing voters for First-Last is equivalent to EPBM and so it is unlikely to be NP-complete, and showing that it is in P would solve an important open question in matching. Linking the complexity of a problem to Exact Perfect [Bipartite] Matching and showing that the problem is in RP can be a useful approach for gaining insight into the complexity of a voting problem which has resisted classification. In fact we find an RP (and RNC) upper-bound for the complexity of 2-Approval-CCRV by showing that it logspace dtt reduces to Exact Perfect Matching (Corollary 11). It remains open if a reduction exists in the other direction. Notably, 2-Approval-CCRV is the sole remaining case in the 11-by-12 table from Erdélyi et al. [2021], and we show that this problem is easy, since it is in RP and it is widely assumed that $\mathrm{P} = \mathrm{RP}$.

As was the case with First-Last, control by adding voters for 2-Approval elections is in P [Lin, 2011]. The corresponding case of exact control is also in P, by a straightforward reduction (for details see the full version).

Theorem 10 *2-Approval-CCAV! is in* P.

We now turn to the problem of control by replacing voters. We show that 2-Approval-CCRV! is equivalent to Exact Perfect Matching through the following cycle of reductions.

1. 2-Approval-CCRV! logspace many-one reduces to Exact Red-Blue Perfect b-Matching (Theorem 12).
2. Exact Red-Blue Perfect b-Matching logspace many-one reduces to Exact Perfect Matching (Theorem 9).
3. Exact Perfect Matching logspace many-one reduces to 2-Approval-CCRV! (Theorem 13).

As a result of Observation 6 and items 1 and 2 we have the following corollary.

Corollary 11 *2-Approval-CCRV logspace dtt reduces to Exact Perfect Matching.*

This implies that 2-Approval-CCRV is in RNC. Additionally, the above statements imply 2-Approval-CCRV! is in RNC.

Theorem 12 *2-Approval-CCRV! logspace many-one reduces to Exact Red-Blue Perfect b-Matching.*

Proof. Consider an instance of 2-Approval-CCRV! with candidates C, registered voters X, unregistered voters Y, preferred candidate p, and number ℓ. As was done in the proof of Theorem 8, we can view this problem as a version of exact control by adding voters where we ask if it is possible to add (to the empty set) $k = \|X\| - \ell$ voters from X and ℓ voters from Y such that p wins.

Note we can assume that we will add as many voters that approve of p as possible. This fixes the final score of p which we denote by $s_p = \min(k, \#\text{voters in } X \text{ approving } p) + \min(\ell, \#\text{voters in } Y \text{ approving } p)$.

We will now construct the instance of Exact Red-Blue Perfect b-Matching. Let the set of vertices in the constructed graph $V(G) = C \cup \{x\}$, where x is a vertex used to ensure that the resulting matching is perfect. For each $c \in C$ let $b(c) = s_p$. Let $b(x) = \|C\|s_p - 2(k + \ell)$. If this is negative, control is not possible. The set of edges is defined as follows.

Red edges: For each voter in X that votes $\{a, b\} > \dots$ add a red edge (a, b).

Blue edges: For each voter in Y that votes $\{a, b\} > \dots$ add a blue edge (a, b).

Uncolored edges: For each candidate $c \in C - \{p\}$, add $b(c) = s_p$ uncolored edges (c, x).

Suppose there exists a way to add k voters from X and ℓ voters from Y such that p wins. As mentioned above, we can assume that we add as many voters approving p as possible and so the score of p is s_p.

We construct a matching in the graph G by taking the k red edges corresponding to the voters added from X and the ℓ blue edges corresponding to the voters added from Y. Since the score of p is s_p, vertex p is incident to $b(p)$ edges in the matching. Since p is a winner, every other candidate c has score at most s_p, and so vertex c is incident to at most s_p edges in the matching. What remains is to make this a perfect matching. The sum of the b-values for the vertices in C is $(\|C\| - 1)s_p$. Since there are $k + \ell$ edges in the matching and the vertex p is an endpoint of s_p of the edges, $(\|C\| - 1)s_p - (2(k + \ell) - s_p)$ uncolored edges must be added. Notice that is exactly $b(x)$ and so this is a perfect matching.

For the converse, suppose there exists a perfect matching with k red edges and ℓ blue edges. For each of the k red edges add the corresponding voter from X, and for each of the ℓ blue edges add the corresponding voter from Y. Since this is a perfect b-matching, vertex p is incident to exactly s_p colored edges in the matching. And each vertex $c \in C - \{p\}$ is incident to at most s_p colored edges in the matching. It is easy to see that when the voters in X are added that correspond to the red edges and the voters in Y are added that correspond to the blue edges that p wins. ❑

What is left to show for equivalence is the reduction in the other direction. The proof of the following theorem can be found in the full version.

Theorem 13 *Exact Perfect Matching logspace many-one reduces to 2-Approval-CCRV!*

5 Conclusion

We showed that by considering RP as an option, we can gain insight into the complexity of voting problems that have resisted classification as being in P or NP-complete. We connected the complexity of control by replacing voters for First-Last and 2-Approval elections to the complexity of the Exact Perfect [Bipartite] Matching, showing these problems in RP. For 2-Approval control by replacing voters, this shows the last remaining case in the comprehensive table from Erdélyi et al. [2021] to be easy, since it is widely assumed $\mathrm{P} = \mathrm{RP}$. We expect this approach will be useful in exploring the complexity of other voting problems.

Acknowledgements

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

References

[Bartholdi *et al.*, 1992] 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.

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

[Dorn and Schlotter, 2017] B. Dorn and I. Schlotter. Having a hard time? Explore parameterized complexity! In U. Endriss, editor, *Trends in Computational Social Choice*, chapter 11, pages 209–230. AI Access, 2017.

[Erdélyi *et al.*, 2021] 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.

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

[Fenner *et al.*, 2021] S. Fenner, R. Gurjar, and T. Thierauf. Bipartite perfect matching is in quasi-NC. *SIAM Journal on Computing*, 50(3):STOC16-218–STOC16-235, 2021.

[Fitzsimmons and Hemaspaandra, 2022] Z. Fitzsimmons and E. Hemaspaandra. Insight into voting problem complexity using randomized classes. Tech. Rep. arXiv:2204.12856 [cs.GT], arXiv.org, April 2022.

[Gill, 1977] J. Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6(4):675–695, 1977.

[Giorgos, 2019] P. Giorgos. Voting rules for expressing conditional preferences in multiwinner elections. Master's thesis, National Technical University of Athens, 2019.

[Hemaspaandra *et al.*, 1993] L. Hemaspaandra, S. Jain, and N. Vereshchagin. Banishing robust Turing completeness. *International Journal of Foundations of Computer Science*, 4(3):245–265, 1993.

[Hemaspaandra *et al.*, 2014] E. Hemaspaandra, L. Hemaspaandra, and H. Schnoor. A control dichotomy for pure scoring rules. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 712–720, July 2014.

[Lin, 2011] A. Lin. The complexity of manipulating k-approval elections. In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence*, pages 212–218, January 2011.

[Loreggia *et al.*, 2015] 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*, pages 1737–1738, May 2015.

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

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

[Papadimitriou, 1994] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Rothe and Schend, 2013] J. Rothe and L. Schend. Challenges to complexity shields that are supposed to protect elections against manipulation and control: A survey. *Annals of Mathematics and Artificial Intelligence*, 68(1–3):161–193, 2013.

[Sipser, 1982] M. Sipser. On relativization and the existence of complete sets. In *Proceedings of the 9th International Colloquium on Automata, Languages, and Programming*, pages 523–531, July 1982.