\textbf{X Then X: Manipulation of Same-System Runoff Elections}\textsuperscript{*}

Zack Fitzsimmons, Edith Hemaspaandra
Department of Computer Science
Rochester Institute of Technology
Rochester, NY 14623, USA

Lane A. Hemaspaandra
Department of Computer Science
University of Rochester
Rochester, NY 14627, USA

January 25, 2013

Abstract

Do runoff elections, using the same voting rule as the initial election but just on the winning candidates, increase or decrease the complexity of manipulation? Does allowing revoting in the runoff increase or decrease the complexity relative to just having a runoff without revoting? We show that, for both weighted and unweighted voting, every possible answer can be made to hold, even for election systems with simple winner problems: The complexity of manipulation, manipulation with runoffs, and manipulation with revoting runoffs are completely independent, in the abstract. On the other hand, for some important, well-known election systems we determine what holds for each of these cases. For no such systems do we find runoffs lowering complexity, and for some we find that runoffs raise complexity.

1 Introduction

There is an extensive literature on two-stage and multistage voting. Although some of this study exists within economics, multistage elections and runoffs have been greatly influential in computational social choice during the past decade, due to such work as that of Elkind and Lipmaa \cite{Elkind2005} and Conitzer and Sandholm \cite{Conitzer2006}.

Particularly interesting recent work in this line has been done by Narodytska and Walsh \cite{Narodytska2012}. They focus on manipulation of election systems of the form \textsc{X Then Y}, i.e., an initial-round election under voting rule \textsc{X}, after which if there are multiple winners just those winners go on to a runoff election under voting rule \textsc{Y}, with the initial votes now restricted to the remaining candidates. The question at issue is whether a given manipulative coalition can vote in such a way as to make a distinguished candidate win (namely, win in the initial round if there is a unique winner in the initial round, or if not, then be a winner of the runoff).

\textsuperscript{*}Supported in part by grants NSF-CCF-\{0915792,1101452,1101479\}.\textsuperscript{1}
Narodytska and Walsh [2012] study the computational complexity of this question. They strongly address the issue of how the manipulation complexity of \( X \) and \( Y \) affect the manipulation complexity of \( X \text{ Then } Y \). Viewing \( P \) as being easy and NP-hardness as being hard, they show that every possible combination of these manipulation complexities can be achieved for \( X \), \( Y \), and \( X \text{ Then } Y \).

The present paper focuses on the complexity of \( X \text{ Then } X \). That is, we are focused on the case where \( X \) is so valued as an election system that if \( X \) selects a unique winner, our election is over and we have our winner. However, if \( X \) in the initial round has tied winners, then we take just those winners and subject them to a runoff election, again using system \( X \). (Votes in this second election will be over only the candidates who made it to the second round.) We are interested in the case in which the second-round votes are simply the initial-round votes restricted to the remaining candidates, and the case in which revoting is allowed in the second round.

Real-world examples exist of such same-system runoff elections. In general elections in North Carolina and many districts of California, election law specifies that if there are two or more candidates tied for being the winner in the initial plurality election, a plurality runoff election is held among just those candidates [Nor 2013, Cal 2013]. So (Plurality THEN Plurality)-with-revoting is being used.

Although Narodytska and Walsh [2012] for \( X \text{ Then } Y \) elections showed that all combinations of \( P \) and NP-hardness for \( X \), \( Y \), and \( X \text{ Then } Y \) can be realized, their examples achieving that almost all have \( X \neq Y \). Thus their broad results do not address the issue of whether all possibilities can be achieved if one seeks to use the same system for both the initial and the runoff election.

We show that every possibility can be achieved, even when the runoff is the same system as the initial election. Indeed, even in the three-way comparison of the complexity of \( X \), the complexity of \( X \) with runoff (under \( X \)), and the complexity of \( X \) with a runoff (under \( X \)) with revoting, we show that every possibility of setting some or all of those to \( P \) or to NP-complete manipulation complexities can be realized. And we show that that can even be done while ensuring that the winner problem for \( X \) (i.e., determining whether a given candidate is a winner of a given election under \( X \)) remains in \( P \), and can also be done both for the weighted and the unweighted cases. For example, there are election systems \( X \)—having \( P \) winner problems—such that manipulation of \( X \) is NP-complete, manipulation of \( X \text{ Then } X \) is NP-complete, but manipulation of \( X \text{ Then } X \) with revoting is in \( P \). And there are election systems \( X \)—having \( P \) winner problems—such that manipulation of \( X \) is in \( P \), manipulation of \( X \text{ Then } X \) is NP-complete, but manipulation of \( X \text{ Then } X \) with revoting is in \( P \). Briefly put, there is no inherent connection between these three complexities.

For the most important systems, however, it is very important to see what the effects of runoffs, and revoting runoffs are. For example, weighted plurality is easily seen to remain easy in all of our cases, e.g., manipulation of elections with runoffs, or with revoting runoffs, remains in \( P \). However, that result itself is something of a fluke. We show that for every (so-called) scoring protocol that is not Triviality, Plurality, or a disguised version of one of
those, manipulation of elections with runoffs and manipulation of elections with revoting runoffs are NP-complete. Although manipulation of unweighted veto is in P, we show that manipulation of unweighted veto elections with runoffs and manipulation of unweighted veto elections with revoting runoffs are NP-complete. For unweighted HalfApproval (the scoring protocol where each voter gives one point to his or her ⌈∥C∥/2⌉ top candidates and zero points to the rest), we prove that for both elections with runoffs and elections with revoting runoffs, the manipulation complexity, even when restricted to having at most one manipulator, is NP-complete. This contrasts with the nonrunoff manipulation complexity here when there is one manipulator, which clearly is P.

For the case of one manipulator, a standard way of seeking to manipulate unweighted or weighted scoring protocols—pioneered for the unweighted case by Bartholdi, Tovey, and Trick [1989a], and extended in many papers since—is to use the natural greedy algorithm. However, we prove that for some scoring protocols X, the greedy approach fails on X THEN X.

2 Related Work

There are quite a few papers whose focus is close to ours. Yet each differs in some important way.

Centrally underpinning our study and framing is the creative, direction-opening work of Narodytska and Walsh [2012] on manipulating X THEN Y elections. (Indeed, their paper even proposes the study of revoting in the second round, and—although in contrast with the present paper they do not seek complexity results regarding revoting—they give a convincing example of why that can make a difference in what can be manipulated.) In a very real sense, our paper is merely about their diagonal—the case when one uses the same election in the original election and the runoff. However, since they were not specifically exploring the diagonals, their existence results in general don’t address that case. However, we must mention an important exception. They show that for STV′, a particular decisive form of STV, that STV and STV′ THEN STV′ are both NP-hard. Our constructions, which must work within a single system for both rounds, are quite different from theirs.

1To avoid confusing the literature’s terminology, it is important for us to mention that there is a very slight, but arguably philosophically interesting, difference between the THEN we defined in the Introduction and the THEN operator as defined by Narodytska and Walsh [2012]. Our and their definitions of X THEN Y can differ in outcome only on what happens if there is exactly one winner of the initial election. In our use of THEN (as given in this paper), in that decisive case the election is over. In their case, that one winner goes on to a one-person election under system Y. Their approach opens the door to having system Y in some cases kill off a single candidate who won the initial round. However, we stress that in their paper they absolutely never use that possibility, and so every result in their paper, including each one mentioned in this paper, holds equally well in both models. Indeed, for any election system that always has at least one winner when there is at least one candidate, the two models coincide, and almost all natural election systems have this property. Nonetheless, to avoid causing any confusion as to terminology, we will henceforward avoid using the term THEN, and will generally speak of elections “with runoff” or “with revoting runoff,” to refer to the cases we here are considering.
In contrast, the work of Elkind and Lipmaa [2005] has a section on using the same system in each round, which is our focus also. However, their model (unlike Narodytska and Walsh and unlike our paper, which pass forward just the winners) is based on removing only the least successful candidate after a round. In particular, their model is of one or more initial rounds, that use a “prune off the last successful candidate” (although in one case they prune off half the candidates) rule inspired by some election system $X$, after which there is a final round using some election system $Y$. So their section on using the same system is about having one or more rounds using (a variant of) $X$ to cut off the least popular candidate, and then a final round also using $X$. Other recent work on removing weakest candidates, usually sequentially, include that of Bag, Sabourian, and Winter [2009] and Davies, Narodytska, and Walsh [2012].

Related to the Elkind–Lipmaa work is the “universal tweaks” work of Conitzer and Sandholm [2006], which shows that adding one pairwise (so-called) CUP-like “preround,” which cuts out about half the candidates, can tremendously boost a system’s manipulation complexity over a broad range of systems.

Speaking more broadly, the problem that Narodytska and Walsh [2012] and this paper are studying, for the case of runoffs and runoffs with revoting, is the manipulation problem. This asks whether a coalition of manipulators can ensure that a particular candidate is a winner of the overall election. The seminal work on the computational complexity of manipulation was that of Bartholdi, Tovey, and Trick [1989a] and Bartholdi and Orlin [1991], and there have been many papers since studying manipulation algorithms for, and hardness results for, a variety of election systems, see, e.g., the survey Faliszewski et al. [2009]. This entire stream exists within the area known as computational social choice Chevaleyre et al. [2007].

Finally, we mention that there is an interesting line of work of Meir et al. [2010] and Lev and Rosenschein [2012] studying in a fully game-theoretic setting iterated voting, in the sense of seeing whether a Nash equilibrium is reached. This work does not remove candidates after votes, and so is different in flavor and goal from our work.

3 Preliminaries

Each election instance will have a finite set, $C$, of candidates, e.g., a particular election might have Obama and Romney as its candidates. Elections also have a finite collection of votes, which we will assume are input as a list of ballots, one per voter. Although social choice theory sometimes allows voters to have names, in this paper we study the most natural case—the one where votes come in nameless, and the election system’s outcome depends on just what the multiset of votes is. We will refer to the collection of votes as $V$.

The type of each vote will depend on the election system. Most systems require a tie-free linear ordering of the candidates, and that will be the case for all systems discussed in this paper.

So-called scoring protocols such as Plurality, Veto, Borda, and so on will for us have votes cast as linear orders. And then from those orders we will assign points to each candidate.
based on the rules of that scoring system. For example, in a veto election, each voter casts zero points for his or her least favorite candidate, and one point for each other candidate. In a plurality election, each voter casts one point for his or her favorite candidate, and zero points for each other candidate. In HalfApproval, if there are \( m \) candidates, each voter gives one point to each of the \( \lceil m/2 \rceil \) top candidates in his or her linear order, and gives zero points to each other candidate. In any scoring system, all points for each candidate are added up, and the candidate(s) who have the maximum score achieved by any candidate are the winner(s). (When we speak of scoring protocols in the abstract, each scoring protocol must have a fixed number of candidates. However, when we say Plurality or HalfApproval or so on, we usually are referring to the protocol that on \( m \)-candidate inputs uses the \( m \)-candidate Plurality or HalfApproval or so on scoring protocol mentioned above.)

An election system, \( X \), is a mapping that given \( C \) and \( V \) outputs a member of the power set of \( C \); the member(s) of that output set are said to be the winner(s). This is precisely the definition of a social choice correspondence, as given in Shoham and Leyton-Brown [2009].

The winner problem for an election system \( X \) is the language that contains exactly those triples \( C, V, p \in C \) such that \( p \) is a winner in the \( X \) election on \( C \) and \( V \). Although some well-known election systems exist whose winner problems are not in P [Bartholdi et al. [1989]], all the systems we study in this paper have P winner problems.

We now define the classic unweighted and weighted election manipulation problems, respectively due to Bartholdi, Tovey, and Trick [1989a] and Conitzer, Sandholm, and Lang [2007]. The unweighted version, called Constructive Unweighted Coalitional Manipulation (CUCM), is defined as follows for any given election system \( X \).

**Name:** \( X \)-CUCM.

**Given:** A set \( C \) of candidates, a collection \( V_1 \) of the nonmanipulative votes (each specified by a tie-free linear ordering over the candidates), a set \( V_2 \) of manipulative voters (since our voters do not have names, these are specified by a nonnegative integer input in unary giving the number of manipulative voters), and a distinguished candidate \( p \in C \).

**Question:** Is there a way to set the votes of the manipulators, \( V_2 \), so that under the election system \( X \), \( p \) is a winner of the election over candidate set \( C \) with the vote set being the ballots of the manipulators and the nonmanipulators?

The analogous weighted version, \( X \)-CWCM, is the same except each member of \( V_1 \) has both a weight and a tie-free linear order, and \( V_2 \) is specified as a list giving the weight of each manipulator. The allowed range of weights is the positive integers.

---

2That definition and this paper allow, as do many papers in computational social choice theory, the case in which an election has no winners. We find that natural for symmetry with the case in which everyone wins. Also, there are real-world cases in which having no winner is natural, e.g., the system for electing players to the Baseball Hall of Fame is set up so that if the crop of candidates in a given year is weak no one will win. That has happened four times, most recently in the January 2013 vote, in which none of the 37 candidates were elected to the Hall.
Our interest here is in runoff elections. So in addition to the above classic versions, let us define versions with runoffs and with revoting runoffs. The “runoff” problems $X$-CUCM-runoff and $X$-CWCM-runoff are the same as the above problems, except if after the $X$ election there are two or more winners, a runoff election is conducted under $X$, with the candidates being just the winners of the initial election, and the votes of all voters (both manipulators and nonmanipulators) being their initial-election’s preference-order vote, restricted to the remaining set of candidates. The “revoting runoff” (or for short, “revoting”) problems $X$-CUCM-revoting and $X$-CWCM-revoting are the same as the above runoff problems, except if there is a runoff election, the manipulators may change their votes. And the question is, of course, whether in this setting there is a set of initial-round and, if needed, second-round manipulator votes that makes $p$ a winner of the overall election.

Note that all of these problems are defined as language problems, as is standard in the area. Typical complexities that they might take on are membership in P and NP-completeness. Those two cases are the focus of this paper and of most papers in this area. However, we mention in passing three related issues. First, it has recently been pointed out that at least in some artificial cases, election decision problems can be in P even when their related search problems are NP-hard. This worry does not infect any of this paper’s results. Every result where we make a polynomial-time claim in this paper has the property that in polynomial time one can even produce the action(s) that achieve the desired outcome (such as making the given candidate win), i.e., our polynomial-time results are essentially what is sometimes called “certifiable,” see Hemaspaandra, Hemaspaandra, and Rothe.

Second, and on the other end of the complexity range, there has been much worry about, and some empirical studies suggesting, that perhaps even NP-complete sets can be often easy. Only during the past half decade has computer science obtained the following remarkably strong result showing that this cannot happen: If even one NP-complete set has a (deterministic) polynomial-time heuristic algorithm whose asymptotic error frequency is subexponential, then the polynomial hierarchy collapses. See the expository article of Hemaspaandra and Williams for a discussion of that result and an attempt to reconcile it with the good empirical results observed for hard problems. (Even for election problems, heuristics seem to often do very well, as shown in a number of papers by Walsh and his collaborators, see, e.g., Walsh.) Our view is that the issue of proving rigorous results about the performance of heuristics on election problems is a highly difficult, highly important direction, but that NP-completeness results for a given problem are un-

---

3For the case of revoting runoffs, the natural model here, in terms of seeking a polynomial-time certificate- (i.e., action-) yielding algorithms, is to allow the manipulative coalition, before the runoff election, a full view of all the initial votes and candidates, and of the outcome of that election, and to require that they set their votes in polynomial time, and of course to also require that their initial-election vote-setting be done in polynomial time. However, since all the election systems in this paper have polynomial-time winner problems, after a given set of initial-round votes the manipulators can themselves compute who the initial-round winner(s) are, and so for problems with p-time winner algorithms, one can w.l.o.g. require the manipulative coalition to fork over at the same time both of its rounds of votes.
questionably an excellent indication that p-time algorithms, and even p-time heuristics with subexponential error rates, cannot be reasonably expected. Thirdly, we mention that in our model, as is standard in this area, the manipulators are given access to the votes of the nonmanipulators. This is a strong though standard assumption, and admittedly is a model for study rather than a perfect image of the real world. The model actually makes the NP-hardness results stronger (since they say that even with full information the problem remains intractable) and most of our results are NP-hardness results.

4 Results

We now turn to our results regarding the complexity of the manipulation problem for elections, for elections with runoffs, and for elections with revoting runoffs.

Our results are of two basic sorts. First, we are interested in what can happen. That is, for those three manipulation complexities, what is the relationship between them? Is there any connection at all?

We show that there is no connection that holds globally. Even when limiting ourselves just to election systems with P winner problems, we prove that every possible case of P-or-NP-complete can simultaneously hold for these three complexities: Each of the 8 weighted and 8 unweighted possibilities can be realized. The reason we want to know what can happen is because it is important to know the universe of behaviors that one may face. Note that since our runoff and revoting problems must have the same system used in the initial and runoff rounds, the result we mention does not follow from the important work of Narodytska and Walsh [2012] realizing all possibilities for $X \Rightarrow Y$; also, they did not look at the complexity of revoting, although as mentioned earlier they did identify and commend revoting as an important area for study.

Our second type of result regards what does happen for the most famous, important, natural systems. For example, although we show that, perhaps counterintuitively, runoffs and revoting runoffs can sometimes lower complexity and can have other bizarre relative complexities, for none of the natural, concrete systems we have looked at do we find this behavior to occur. For each concrete, natural system we have studied, runoffs and revoting runoffs either leave the manipulation complexity unchanged, or increase the manipulation complexity. Of course, our results on what does happen for concrete systems prove some of the cases of our claims regarding what can happen.

The following theorem states our result about what can happen, namely, regarding P and NP-completeness, any possible triple of complexities can occur.

**Theorem 4.1** Let NPC denote “NP-complete.” Let $W = \{(P, P, P), (P, P, NPC), (P, NPC, P), (P, NPC, NPC), (NPC, P, P), (NPC, P, NPC), (NPC, NPC, P), (NPC, NPC, NPC)\}.$

1. For each element $w$ of $W$, there exists an election system $X$, whose winner problem is in $P$, such that the complexity of $X$-CUCM, $X$-CUCM-runoff, and $X$-CUCM-revoting is, respectively, the three fields of $w$. 
2. The analogous result holds for the weighted case (where the three fields will capture the complexity of $X$-CWCM, $X$-CWCM-runoff, and $X$-CWCM-revoting, respectively).

We in the rest of this report will present our results about concrete systems, and some proofs/sketches regarding our results on those. (However, we mention—and an extended version of this report will give complete details on all the constructions—that some of the more counterintuitive cases within the above theorem involve novel proof approaches. The two most interesting unweighted cases are the ones realizing the cases (NPC, P, NPC) and, especially, (NPC, NPC, P). The key twist in these is that both create a setting in which an election system can in effect pass messages to its own second-round self through the winner set and with the help of the manipulators. In particular, in a certain set of circumstances, the election system can be made to, in effect, know that “If the input I’m seeing is taking place in a second round (although I cannot myself tell whether or not it is), then we are utterly certainly in a model in which revoting is allowed and indeed in which one of the manipulators has changed his or her vote since the initial round.”)

The following result provides an unweighted case where the classic manipulation problem is simple but the runoff and revoting runoff versions are hard. To support this contrast, we must mention that it is well-known that Veto-CUCM is in P.

**Theorem 4.2** Veto-CUCM-runoff and Veto-CUCM-revoting are each NP-complete.

**Proof.** We will reduce from from the well-known NP-complete Exact Cover by 3-Sets Problem (X3C): Given a set $B = \{b_1, \ldots, b_{3k}\}$, and a collection $S = \{S_1, \ldots, S_n\}$ of 3-element subsets of $B$, we ask if $S$ has an exact cover for $B$, i.e., if there exists a subcollection $S'$ of $S$ such that every element of $B$ occurs in exactly member of $S'$. Without loss of generality, we assume that $n \geq 3$. We will denote which elements of $B$ are in a given $S_i$ by some new $b_{ij}$ variables: $S_i = \{b_{i1}, b_{i2}, b_{i3}\}$.

Since Veto-CUCM is in P (simply greedily veto all candidates that score higher than $p$), the only place where hardness can come in is in the selection of the set of winners in the initial round.

Our election has the following candidates: $p$ (the preferred candidate), $b_1, \ldots, b_{3k}$ and $s_1, \ldots, s_n$ (candidates corresponding to the X3C instance), $r_1, \ldots, r_k$ (candidates that will be vetoed in the runoff), $d$ (a buffer candidate), and $\ell$ (a candidate that always loses in the initial round). We have $k$ manipulators. We have the following nonmanipulators:

- For every $i, 1 \leq i \leq n$, one nonmanipulator voting
  $$\ldots > p > b_{i1} > s_i$$
  ($\ldots$ denotes that the remaining candidates are in arbitrary order).

- For every $i, 1 \leq i \leq n$, one nonmanipulator voting
  $$\ldots > p > b_{i2} > s_i.$$

- For every $i, 1 \leq i \leq n$, one nonmanipulator voting
  $$\ldots > p > b_{i3} > s_i.$$
• Three nonmanipulators voting \( \cdots > p \).

• For every \( c \in B \cup \{r_1, \ldots, r_k\} \cup \{d\} \), three nonmanipulators voting \( \cdots > p > c \).

• One nonmanipulator voting \( \cdots > p > \ell \).

• For every \( i, 1 \leq i \leq n \), one nonmanipulator voting \( \cdots > p > d > s_i > \ell \).

Note that every candidate other that \( \ell \) receives 3 vetoes from the nonmanipulators in the initial round.

Let \( S' = \{S_{j_1}, \ldots, S_{j_k}\} \) be an exact cover for \( S \). For \( 1 \leq i \leq k \), let the \( i \)th manipulator vote \( \cdots > r_i > s_{j_i} \). We claim that \( p \) is a winner of the overall election (even without revoting). It is immediate that the winner set of the initial round is \( C - \{\ell\} - \{s_j \mid S_j \in S\} \). Since \( \ell \) does not participate in the runoff, \( p \) gains one veto from the nonmanipulator voting \( \cdots > p > \ell \) and each \( s_i \) that participates in the runoff gains one veto from the nonmanipulator voting \( \cdots > d > s_i > \ell \). \( d \) gains \( k \) vetoes from the nonmanipulators voting \( \cdots > d > s_i > \ell \) such that \( S_i \in S' \) and every \( b \in B \) gains one veto from the nonmanipulator voting \( \cdots > p > b > s_i \) such that \( b \in S_i \) and \( S_i \in S' \). Every candidate \( r_i \) gains a veto from the manipulator voting \( \cdots > r_i > s_{j_i} \). It follows that \( p \) is a winner of the runoff.

For the converse, we will show the manipulations described above are the only way to make \( p \) a winner. Suppose the manipulators can vote (in the initial round and the runoff) in such a way that \( p \) becomes a winner of the overall election. Recall that in the initial round, every candidate other that \( \ell \) receives 3 vetoes from the nonmanipulators and that \( \ell \) receives \( n + 1 \) vetoes. Since \( n \geq 3 \) and there are \( k \) manipulators, \( \ell \) does not participate in the second round and at most \( k \) other candidates (the ones vetoed by a manipulator) do not participate in the second round. Since \( \ell \) does not participate in the second round, \( p \) gains one veto from the nonmanipulator voting \( \cdots > p > \ell \).

Suppose there is a candidate \( c \in B \cup \{r_1, \ldots, r_k\} \cup \{d\} \) that does not participate in the second round of the election. Then \( p \) gains 3 vetoes from the nonmanipulators voting \( \cdots > p > c \), and thus \( p \) receives at least 7 vetoes in the second round. There are at least \( 2k \) candidates from \( B \) that participate in the second round and each of these candidates is vetoed 3 times in the initial round and does not gain any vetoes from deleting \( \ell \). Since \( p \) receives at least 7 vetoes in the second round, each candidate in \( B \) that participates in the second round needs to gain at least 4 vetoes, so these candidates need to gain a total of at least \( 8k \) vetoes. But the most vetoes that these candidates can gain is 3 vetoes for each candidate \( s_i \) that does not participate in the second round plus \( k \) vetoes from the manipulators. Since fewer than \( k \) \( s_i \) candidates do not participate in the runoff, the \( B \) candidates that participate in the runoff gain a total of at most \( 4k \) vetoes, which is not enough.

It follows that the only candidates other than \( \ell \) that do not participate in the second round are \( s_i \) candidates. Note that candidates in \( \{r_1, \ldots, r_k\} \) will not gain vetoes from the nonmanipulators, and so each manipulator needs to veto exactly one \( r_i \). To make sure that every candidate \( b \in B \) gains at least one veto, we need to delete a set of \( s_i \) candidates.
corresponding to a cover. Since we can delete at most \( k \) such candidates, these candidates will correspond to an exact cover.

It is easy to argue, in contrast with the result of Theorem 4.2 regarding Plurality’s close cousin Veto, that Plurality is easy, even in the weighted case, since throwing all one’s votes to \( p \) is always optimal.

**Theorem 4.3** Plurality-CWCM-runoff and Plurality-CWCM-revoting are each in \( P \).

We mention the following result, which holds because by brute-force partitioning of the integer \( \|V\| \) into at most \( (\|C\|)!)^2 \) named buckets (one for each pair of possible votes, though a second-round decrease in candidates could make the numbers even smaller than this), one can solve even the revoting runoff manipulation question (and of course the same holds for plain runoffs).

**Theorem 4.4** For any election system \( X \) having a \( P \) winner problem, and for any integer \( k \), \( X \)-CUCM-revoting restricted to \( k \) candidates is in \( P \).

The following claim transfers to our two problems the dichotomy result for scoring protocols known for the nonrunoff case.

**Theorem 4.5** For every weighted scoring protocol \( X \), \( X \)-CUCM-runoff and \( X \)-CUCM-revoting are in \( P \) if \( X \) is Plurality or Triviality (or a direct transform of one of those, in a sense that can be made formal, see Hemaspaandra and Hemaspaandra [2007]), and otherwise are \( \text{NP-complete} \).

**Proof.** For Plurality, this follows from Theorem 4.3 and for Triviality, this is trivial. For every other weighted scoring protocol \( X \), Hemaspaandra and Hemaspaandra [2007] give a reduction \( f \) from the \( \text{NP-complete} \) problem Partition to \( X \)-CUCM with the property that for all \( x \), if \( x \in \text{Partition} \), then \( p \) can be made the unique winner in \( f(x) \), and if \( x \notin \text{Partition} \), then \( p \) can not be made a winner in \( f(x) \). So, if \( x \in \text{Partition} \), then \( x \) can be made the unique winner of the initial round, and thus the unique winner of the overall election. And if \( x \notin \text{Partition} \), then \( p \) will never make it to the final round.

The case of just one manipulator is a natural and important case. It also can often be surprisingly well handled, thanks to the lovely result—initially due for the unweighted case to the seminal work of Bartholdi, Tovey, and Trick [1989a] and since then much extended—that the natural (p-time) greedy manipulation algorithm (giving one’s highest point value to \( p \) and then giving, in turn, the highest remaining value to the candidate who has the lowest point total among those not yet assigned points by the manipulative voter) is optimal (i.e., finds a successful manipulation when one exists) for both weighted and unweighted scoring protocols, for the case when there is just one manipulator. The following theorem states that that result does not carry over to runoff elections.
Theorem 4.6 The standard 1-manipulator p-time greedy algorithm for scoring protocols is not optimal for X-CUCM-runoff or X-CUCM-revoting, restricted to at most one manipulator, where X is the family of scoring protocols (2, 1, 0, ..., 0).

Proof. Consider the election with candidate set \{p, a, b, c\}, two nonmanipulators voting \(a > p > c > b\) and \(b > a > c > p\), and one manipulator. The scores of \(p, a, b, c\) from the nonmanipulators are 1, 3, 2, 0. The greedy algorithm would give the following vote for the manipulator: \(p > c > b > a\). Then \(p\) and \(a\) are the winners of the initial round, and there is no way for \(p\) to win the second round. However, if the manipulator votes \(p > b > c > a\), then \(p, a,\) and \(b\) are the winners of the initial round, and \(p\) is a winner of the runoff (even without revoting). 

Theorem 4.2 gave a case where a simple-to-manipulate unweighted scoring protocol became hard for runoffs, with or without revoting. The following result gives a new example of runoffs increasing complexity, this time for the one-manipulator case. It is natural to wonder whether the following theorem itself implies the cousin of Theorem 4.6 in which the protocol \((2, 1, 0, \ldots, 0)\) is replaced by HalfApproval. The answer is that the following theorem does not imply that, but it does imply something a bit weaker than that cousin, namely, it says that that cousin holds unless \(P \neq NP\). (Of course, Theorem 4.6 holds absolutely; it doesn’t require a \(P \neq NP\) hypothesis.) HalfApproval-CUCM for one manipulator is clearly in \(P\)—for example by the greedy algorithm we mentioned above—and so the following result does express a raising of complexity.

Theorem 4.7 HalfApproval-CUCM-runoff and HalfApproval-CUCM-revoting are each NP-complete, even when restricted to having at most one manipulator.

Proof Sketch. We pad the construction from Theorem 4.2. Note that we have fewer candidates in the second round than in the initial round, and so in contrast to the proof of Theorem 4.2 the manipulators have fewer vetoes to contribute in the final round. Without loss of generality, we assume that \(k\) is even. We use the election from the proof of Theorem 4.2 with the following modifications. We delete candidates \(r_{\frac{n}{2}+1}, \ldots, r_k\) (but keep \(r_1, \ldots, r_{\frac{n}{2}}\)). We add candidates \(\hat{r}_1, \ldots, \hat{r}_{n+\frac{3k}{2}+3}\) and for each \(i, 1 \leq i \leq n + \frac{3k}{2} + 3\), we add

- two nonmanipulators voting \(\ldots > p > \hat{r}_i\), and
- one nonmanipulator voting \(\ldots > \hat{r}_i > \ell\).

And we have only one manipulator. Note that we have a total of \(2(n + \frac{3k}{2} + 3)\) candidates. It is easy to see that if \(S' = \{S_{j_1}, \ldots, S_{j_k}\}\) is an exact cover for \(S\), then letting the manipulator vote

\[
\cdots r_1 > \cdots > r_{\frac{n}{2}} > \hat{r}_1 > \cdots > \hat{r}_{n+\frac{3k}{2}+3} > s_{j_1} > \cdots > s_{j_k}
\]

will make \(p\) a winner of the overall election. For the converse, if the manipulator can vote (in the initial round and the runoff) such that \(p\) becomes a winner, note that every \(\hat{r}_i\)
candidate must be vetoed by the manipulator in the initial round and in the second round. This leaves $k$ vetoes for the other candidates in the initial round, and, much like in the proof of Theorem 4.2 it can be shown that those candidates must be $s_i$ candidates that correspond to a cover.

5 Conclusions and Open Problems

This paper has explored the relative manipulation complexity of runoff elections, with and without revoting. We have seen that there is no general relation between the manipulation complexity of either of those with each other or with the manipulation complexity of the underlying election system. Sometimes revoting can even lower complexity, for example. Yet for the natural, concrete systems we studied, runoffs and revoting runoffs never lowered complexity and sometimes raised complexity.

Important open directions include the study of runoffs and revoting runoffs for the case of bribery rather than manipulation, for which we have some preliminary results, and the study of what role heuristics, especially in light of Theorems 4.6 and 4.7 can play.

References

P. Bag, H. Sabourian, and E. Winter. Multi-stage voting, sequential elimination and Condorcet consistency. Journal of Economic Theory, 144(3):1278–1299, 2009.

J. Bartholdi, III and J. Orlin. Single transferable vote resists strategic voting. Social Choice and Welfare, 8(4):341–354, 1991.

J. Bartholdi, III, C. Tovey, and M. Trick. The computational difficulty of manipulating an election. Social Choice and Welfare, 6(3):227–241, 1989a.

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, 1989b.

Tie Votes. California Elections Code, Division 15/Chapter 10/Statute 15650-15673, 2013. http://leginfo.legislature.ca.gov/.

Y. Chevalleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. In Proceedings of the 33rd International Conference on Current Trends in Theory and Practice of Computer Science, pages 51–69. Springer-Verlag Lecture Notes in Computer Science #4362, January 2007.

V. Conitzer and T. Sandholm. Nonexistence of voting rules that are usually hard to manipulate. In Proceedings of the 21st National Conference on Artificial Intelligence, pages 627–634. AAAI Press, July 2006.
V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3):Article 14, 2007.

J. Davies, N. Narodytska, and T. Walsh. Eliminating the weakest link: Making manipulation intractable? In *Proc. of AAAI-12*, pages 1333–1339. AAAI Press, August 2012.

E. Elkind and H. Lipmaa. Hybrid voting protocols and hardness of manipulation. In *Proceedings of the 16th Annual International Symposium on Algorithms and Computation*, pages 206–215. Springer-Verlag *Lecture Notes in Computer Science #3872*, December 2005.

P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. A richer understanding of the complexity of election systems. In *Fundamental Problems in Computing*, pages 375–406. Springer, 2009.

E. Hemaspaandra and L. Hemaspaandra. Dichotomy for voting systems. *Journal of Computer and System Sciences*, 73(1):73–83, 2007.

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

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*. Leibniz International Proceedings in Informatics (LIPIcs), February/March 2013. To appear.

L. Hemaspaandra and R. Williams. An atypical survey of typical-case heuristic algorithms. *SIGACT News*, 43(4):71–89, December 2012.

O. Lev and J. Rosenschein. Convergence of iterative voting. In *Proc. of AAMAS-12*, pages 611–618, June 2012.

R. Meir, M. Polukarov, J. Rosenschein, and N. Jennings. Convergence to equilibria in plurality voting. In *Proc. of AAAI-10*, pages 823–828, July 2010.

N. Narodytska and T. Walsh. Manipulating two stage voting rules. In *Proceedings (Workshop Notes) of the 4th International Workshop on Computational Social Choice*, pages 323–334, September 2012.

*Elections and Election Laws*. North Carolina General Statutes, Chapter 163/Article 15A/Statute 163-182.8, 2013. http://www.ncleg.net/.

Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

T. Walsh. Where are the really hard manipulation problems? The phase transition in manipulating the veto rule. In *Proc. of IJCAI-09*, pages 324–329, July 2009.