Manipulating Two Stage Voting Rules

Nina Narodytska and Toby Walsh

Abstract

We study the computational complexity of computing a manipulation of a two stage voting rule. An example of a two stage voting rule is Black's procedure. The first stage of Black's procedure selects the Condorcet winner if they exist, otherwise the second stage selects the Borda winner. In general, we argue that there is no connection between the computational complexity of manipulating the two stages of such a voting rule and that of the whole. However, we also demonstrate that we can increase the complexity of even a very simple base rule by adding a stage to the front of the base rule. In particular, whilst Plurality is polynomial to manipulate, we show that the two stage rule that selects the Condorcet winner if they exist and otherwise computes the Plurality winner is NP-hard to manipulate with 3 or more candidates, weighted votes and a coalition of manipulators. In fact, with any scoring rule, computing a coalition manipulation of the two stage rule that selects the Condorcet winner if they exist and otherwise applies the scoring rule is NP-hard with 3 or more candidates and weighted votes. It follows that computing a coalition manipulation of Black's procedure is NP-hard with weighted votes. With unweighted votes, we prove that the complexity of manipulating Black's procedure is inherited from the Borda rule that it includes. More specifically, a single manipulator can compute a manipulation of Black's procedure in polynomial time, but computing a manipulation is NP-hard for two manipulators.

1 Introduction

There exist several voting procedures that work in stages. For example, Black's procedure is a two stage voting rule whose first stage elects the Condorcet winner, if one exists, and otherwise moves to a second stage which elects the Borda winner [12]. As a second example, the French presidential elections use a two stage runoff voting system. If there is a majority winner in the first stage, then this candidate is the overall winner, otherwise we go to the second stage where there is a runoff vote between the two candidates with the most votes in the first round. Such two stage voting rules can inherit a number of attractive axiomatic properties from their parts. For example, Black's procedure inherits Condorcet consistency from its first part, and properties like monotonicity, participation and the Condorcet loser property from its second part. Inheriting such properties from its parts might be considered an attractive feature of two stage voting rules. On the other hand, a less desirable property of one of the base rules can infect the overall two stage rule. For instance, it has been shown that, with single peaked votes, many types of control and manipulation problems are polynomial for Black's procedure [4]. This polynomiality is essentially inherited from the first stage of the rule which selects the Condorcet winner (which must exist with single peaked votes). Such vulnerability to manipulation and control might be considered an undesirable property for a two stage voting rule. This raises several interesting questions from the perspective of computational social choice. For example, with unrestricted votes as opposed to single peaked votes, are two stage voting rules more or less computationally difficult to manipulate than single stage voting rules? How does the computational complexity of manipulating a two stage voting rule depend on the computational complexity of manipulating the two rules that it composes? In this paper, we address such questions.

Our work builds upon recent research that looks at methods to combine together voting rules. In [10], we considered a recursive combinator that successively eliminates the least

popular candidate(s). This captures voting rules proposed in the past like those of Nanson, Baldwin or Coombs (all described in more detail in the next section). By comparison, we consider here a sequential combinator where the first rule eliminates all but the most popular candidates and the second rule then decides between those that remain. This captures voting rules proposed in the past like Black's procedure. Perhaps closest to this work is the sequential combinator introduced in [11]. This is an intermediate position between the two extremes of eliminating the least popular and all but the most popular candidates. Elkind and Lipmaa's combinator eliminates candidates by applying some given number of rounds of the first rule before using the second rule to decide between the candidates that remain. Even more recently, we have considered a parallel combinator that combines together the opinions of two (or more) different voting rules [16]. This combinator applies both rules simultaneously and compares their results. As well as proving computational properties of existing voting rules like Black's procedure, this paper strengthens the evidence that adding multiple rounds to voting will often increase the computational resistance to manipulation.

2 Background

A *profile* is a sequence of n total orders over m candidates. A *voting rule* is a function mapping a profile onto a set of *winners* (strictly speaking this is a social choice correspondence). We consider some of the most common voting rules.

Scoring rules: Given a *scoring vector* $(w_1, \ldots, w_m)$ of weights, the ith candidate in a vote scores w_i, and the winner is the candidate with highest total score over all the votes. The **Plurality** rule has the weight vector $(1, 0, \ldots, 0)$, the **Veto** rule has the vector $(1, 1, \ldots, 1, 0)$, and the **Borda** rule has the vector $(m-1, m-2, \ldots, 0)$.

Cup: The winner is the result of a series of pairwise majority elections between candidates. Given the *agenda*, a binary tree in which the roots are labelled with candidates, we label the parent of two nodes by the winner of the pairwise majority election between the two children. The winner is the label of the root.

Black's procedure: This rule has two stages. We first determine if there is a *Condorcet winner*, a candidate that beats all others in pairwise majority comparisons. If there is, this is the winner. Otherwise, we return the result of the *Borda* rule.

Single Transferable Vote (STV): This rule requires up to $m-1$ rounds. In each round, the candidate with the least number of voters ranking them first is eliminated until one of the remaining candidates has a majority.

Nanson's and Baldwin's rules: These are iterated versions of the Borda rule. In Nanson's rule, we compute the Borda scores and eliminate any candidate with less than half the mean score. We repeat until there is an unique winner. In Baldwin's rule, we compute the Borda scores and eliminate the candidate with the lowest score. We again repeat until there is an unique winner.

Coombs' rule: This is an iterated version of the Veto rule. We repeatedly eliminate the candidate with the most vetoes until we have one candidate with a majority.

We consider both unweighted and integer weighted votes. A weighted votes can simply be viewed as a block of identical unweighted votes.

3 Two stage voting rules

We consider a general class of two stage voting rules. Given voting rules X and Y, the rule XTHENY applies the voting Y to the profile constructed by eliminating all but the winning candidates from the voting rule X. Both X and Y can themselves be two stage voting rules giving us the possibility to construct multi-stage voting rules. For example, Black's

procedure is $Condorcet$THEN$Borda$ where $Condorcet$ is the multi-winner rule that elects the Condorcet winner if it exists, and otherwise elects all candidates. As a second example, Plurality with Runoff is $TopTwo$THEN$Majority$ where $TopTwo$ is the multi-winner voting rule that elects the candidates with the two most plurality votes. There are many possible rules that we might choose to combine this way. $Condorcet$ is an attractive choice for the first rule as it guarantees that the resulting combination is Condorcet consistent. However, there are other interesting choices including:

CondorcetLoser: This is the rule that elects all candidates except, when it exists, the Condorcet loser.

CopelandSet: This is the rule that elects all candidates in the Copeland set. The Copeland score of a candidate is the number of candidates that it beats less the number of candidates that beats it. The Copeland set contains those candidates with the maximal Copeland score. When there is a Condorcet winner, this is the only candidate in the Copeland set.

SmithSet: This is the rule that elects all candidates in the Smith set. This is the smallest non-empty set of candidates such that every candidate in the set beats every candidate outside the set in pairwise elections. When there is a Condorcet winner, this is the only candidate in the Smith set. Voting rules like Nanson's and Kemeny are guaranteed to pick candidates from the Smith set.

SchwartzSet: This is the rule that elects all candidates in the Schwartz set. The Schwartz set is a subset of the Smith set and is the union of all the undominated sets. A set is undominated if every candidate inside the set is pairwise unbeaten by every candidate outside, and no non-empty proper subset satisfies this property. When there is a Condorcet winner, this is the only candidate in the Schwartz set.

We can also consider recursive definitions. We suppose any recursion terminates when either we have a single candidate left, or the set of candidates left does not reduce in size. For example, we can recursively define STV by $STV = PluralityLoser$THENSTV where $PluralityLoser$ is the rule that elects all candidates but the candidate with the fewest first place votes. As a second example, we can recursively define Baldwin's rule by $Baldwin = BordaLoser$THEN$Baldwin$ where $BordaLoser$ is the multi-winner rule that elects all candidates but the candidate with the lowest Borda score. Nanson's rule can be defined recursively in a similar way. As a third example, we can define Coombs' rule by $Coombs = Majority$THEN$(VetoLoser$THEN$Coombs)$ where $Majority$ elects the candidate with a majority of first place votes or, if there is no such candidate, elects all candidates, and $VetoLoser$ is the rule that elects all candidates but the candidate with the most last placed votes.

4 Axiomatic and algebraic properties

It is interesting to consider which axiomatic properties are inherited from the base rules being combined. For example, it is simple to see that we can inherit Condorcet consistency or the Condorcet loser properties.

Proposition 1. *For any voting rule X, the combinations $Condorcet$THENX, $CopelandSet$THENX, $SmithSet$THENX and $SchwartzSet$THENX are Condorcet consistent. Similarly, for any voting rule Y, the combination $CondorcerLoser$THENY satisfies the Condorcet loser property.*

With recursively defined rules, we can give a similar result. We say that a multi-winner rule is Condorcet consistent if it includes the Condorcet winner in the set of winners, and satisfies the Condorcet loser property if the set of winners never includes the Condorcet loser.

Proposition 2. *Suppose Y is recursively defined by $Y = X\textsc{Then}Y$ and X is Condorcet consistent. Then Y is also Condorcet consistent. Similarly, if X satisfies the Condorcet loser property then Y does also.*

Note that the Borda loser is never the Condorcet winner. Hence, the multi-winner rule *BordaLoser* is Condorcet consistent. Thus, it follows from Proposition 2 that Baldwin's rule (which is recursively defined using *BordaLoser*) is also Condorcet consistent.

There are also axiomatic properties which can be lost by combining together voting rules. For example, the Borda loser rule which eliminates the lowest Borda scoring candidate is monotonic since increasing one's preference for a candidate can only prevent them from being the Borda loser. However, Baldwin's rule, which is the recursive version of the Borda loser rule, is not monotonic. It will therefore be interesting to identify conditions under which two stage voting rules are monotonic.

This combinator has a number of interesting algebraic properties. For example, the *Identity* rule that returns all candidates is a left and right identity of the THEN combinator. Note that the THEN combinator is neither commutative nor associative. If a voting rule is recursively defined then it is idempotent (that is, $X\textsc{Then}X = X$). More complex algebraic identities can be derived such as the following.

Proposition 3. *If X is idempotent then $X\textsc{Then}(X\textsc{Then}Y) = X\textsc{Then}Y$ and $(Y\textsc{Then}X)\textsc{Then}X = Y\textsc{Then}X$.*

More specialized properties can also be derived such as the following.

Proposition 4. *$SmithSet\textsc{Then}Nanson = Nanson$.*

Proposition 5. *If X is Condorcet consistent and only returns the Condorcet winner when they exist then $Condorcet\textsc{Then}X = X$.*

5 Complexity of manipulation

One of the main contributions of this paper is to consider the impact of two stage voting rules on the computational complexity of computing a manipulation. As in previous studies (e.g. [2, 6]), we consider manipulation with unweighted votes and a small number of manipulators, and manipulation with weighted votes, a coalition of manipulators and a small number of candidates. As is common in the literature, we break ties in favour of the manipulators.

5.1 Weighted votes, general results

With weighted votes, we first argue that is no connection in general between the computational complexity of computing a manipulation of a two stage voting rule and the computational complexity of manipulating its parts.

Proposition 6. *There exist voting rules X and Y with the following properties for weighted votes:*

1. *computing coalition manipulations of X, Y and $X\textsc{Then}Y$ are polynomial;*
2. *computing coalition manipulations of X and Y are polynomial but of $X\textsc{Then}Y$ is NP-hard;*

3. *computing a coalition manipulation of X is polynomial and of Y is NP-hard, but of XTHENY is polynomial;*

4. *computing a coalition manipulation of X is polynomial, but of Y and XTHENY are NP-hard;*

5. *computing a coalition manipulation of X is NP-hard, but of Y and XTHENY are polynomial;*

6. *computing a coalition manipulation of X is NP-hard and of Y is polynomial, but of XTHENY is NP-hard;*

7. *computing coalition manipulations of X and Y are NP-hard but of XTHENY is polynomial;*

8. *computing coalition manipulations of X, Y and XTHENY are NP-hard.*

Proof: The NP-hardness results are derived from the NP-hardness of computing a coalition manipulation of STV with 3 or more candidates [7].

1. Consider $X = FirstRoundCup$ and $Y = Cup$. $FirstRoundCup$ is the multi-winner rule that runs one round of the Cup voting rule. Note that $FirstRoundCup$THENCup is the Cup rule itself, and both $FirstRoundCup$ and Cup are polynomial to manipulate by a coalition even with weighted votes [7].

2. Consider $X = TopTwo$ and $Y = Majority$ where $TopTwo$ elects the two candidates with the two highest plurality scores. On 3 candidates, $TopTwo$THEN$Majority$ is Plurality with runoff, which itself is equivalent STV which is NP-hard to manipulate by a coalition of weighted voters when we have 3 or more candidates [7].

3. Consider $X = Plurality'$ and $Y = STV$ where $Plurality'$ is the decisive form of plurality that includes tie-breaking in some fixed order. Note that XTHENY is again $Plurality'$ which is polynomial to manipulate by a coalition even with weighted votes [7].

4. Consider $X = Identity$ and $Y = STV$ where $Identity$ is the identity rule that elects all the candidates in the election. Note that XTHENY is also STV.

5. Consider $X = STV_1$ which is the multi-winner voting rule that elects both the STV winner and the candidate with the lexicographically smallest label, and Y elects the candidate with the lexicographically smallest label. Note that XTHENY always elects the candidate with the lexicographically smallest label. Such a rule is polynomial to manipulate by a coalition even with weighted votes.

6. Consider $X = STV$ and $Y = Identity$. Note that XTHENY is again STV.

7. Consider $X = STV_2$ and $Y = STV_3$ where STV_2 is the multi-winner rule that elects the STV winner as well as those candidates with the lexicographically smallest and largest names, and STV_3 elects the plurality winner between the candidates with the lexicographically smallest and largest names if there are 3 or fewer candidates and otherwise elects the STV winner. Note that XTHENY elects the plurality winner between the candidates with the lexicographically smallest and largest names, and computing a coalition manipulation of such a rule is polynomial even with weighted votes.

8. Consider $X = Y = STV'$ where STV' is the decisive form of STV where we tie-break in favour of the manipulators. Note that XTHENY is also STV'.

♡

5.2 Weighted votes, specific rules

With weighted votes, we already know that several multi-stage voting rules are NP-hard to manipulate including STV, Plurality with runoff, Baldwin's rule (all with 3 candidates), and Nanson's rule (with 4 candidates) [7, 15]. We first show that computing a manipulation of *Condorcet*THENX with weighted votes is NP-hard for any scoring rule X. This contrasts to scoring rules in general where computing a coalition manipulation is NP-hard for any rule that is not isomorphic to Plurality, but is polynomial for Plurality. This demonstrates that adding the test for a Condorcet winner to give *Condorcet*THENX increases the computational complexity of manipulation over that for the scoring rule X alone.

Proposition 7. *Deciding whether there exists a coalitional manipulation for Condorcet*THEN*Plurality with weighted votes is NP-complete with 3 or more candidates.*

Proof: We reduce from the number partitioning problem with n numbers k_i, $i = 1, \ldots, n$, $\sum_{i=1}^{n} k_i = 2K$. We have n manipulators with the weight k_i each.

Consider a non-manipulator profile. Suppose voters with total weight $2K$ cast (a, b, p) and voters with total weight $2K$ cast (b, a, p). The candidate p is a Condorcet loser as it loses to both a and b. Moreover, as a and b are tied, there is no Condorcet winner.

Note that if all manipulators put p in the first position then p wins under plurality. However, the manipulators have to make sure that they also do not make a or b the Condorcet winner. Note that if a (b) gets a higher score than b (a) then a (b) is the Condorcet winner. Therefore, the only way to prevent one of them from becoming the Condorcet winner is to partition the total weight of votes between a and b. Thus, manipulators with a total weight of K have to vote (p, a, b) and the remaining manipulators have to vote (p, b, a). Therefore, there exists a manipulation iff there is a partition with the required sum K. ♡

Proposition 8. *With weighted votes and any scoring rule X that is not isomorphic to Plurality, computing a coalition manipulation of Condorcet*THENX *is NP-hard for 3 or more candidates.*

Proof: Without loss of generality, we consider a scoring rule which gives a score of α_1 for a candidate in 1st place in a vote, α_2 for 2nd place, and 0 for 3rd place. We adapt the reduction used in the proof of Theorem 6 in [8] for the NP-hardness of manipulating any scoring rule that is not isomorphic to Plurality voting. The reduction is from the number partitioning problem and constructs an election with a weight of $6\alpha_1 K - 2$ votes over the candidates a, b and p (who the manipulators wish to make win). Within these votes, the manipulators have a weight of $2(\alpha_1 + \alpha_2)K$ votes, and the rest are fixed. The number partition problem is to divide a set of integers summing to $2K$ into two equal sums. There is a manipulator of weight k_i for every integer k_i in the set being partitioned. We now add $6\alpha_1 K - 1$ triples of votes: (a, b, p), (b, p, a), (p, a, b). This has no impact on the differences in the scores between the candidates. However, it creates a Condorcet cycle so that there cannot be a Condorcet winner whatever the manipulators do with their votes. Hence, we must pass to the second round where the winner is decided by the scoring rule X. As in the proof of Theorem 6 in [8], there is a manipulation that makes p the winner of the scoring rule X iff there is a partition into two equal sums. Thus, computing a coalition manipulation of *Condorcet*THENX is NP-hard. ♡

It follows immediately that coalition manipulation of Black's procedure, which is *Condorcet*THEN*Borda* is NP-hard with 3 or more candidates.

Corollary 1. *With weighted votes, coalition manipulation of Black's procedure is NP-hard with 3 or more candidates.*

5.3 Unweighted votes, general results

As with weighted votes, there is no connection in general between the computational complexity of computing a manipulation of a two stage voting rule with unweighted votes and the computational complexity of computing a manipulation of its parts.

Proposition 9. *There exist voting rules X and Y with the following properties:*

1. *computing manipulations of X, Y and XTHENY are polynomial;*
2. *computing manipulations of X and Y are polynomial but of XTHENY is NP-hard;*
3. *computing a manipulation of X is polynomial and of Y is NP-hard, but of XTHENY is polynomial;*
4. *computing a manipulation of X is polynomial, but of Y and XTHENY are NP-hard;*
5. *computing a manipulation of X is NP-hard, but of Y and XTHENY are polynomial;*
6. *computing a manipulation of X is NP-hard and of Y is polynomial, but of XTHENY is NP-hard;*
7. *computing manipulations of X and Y are NP-hard but of XTHENY is polynomial;*
8. *computing manipulations of X, Y and XTHENY are NP-hard.*

Proof: The NP-hardness results are derived from the NP-hardness of manipulating STV with unweighted votes and a single manipulator [2].

1 Identical examples to the weighted case.

2 Consider the multi-winner voting rule X that eliminates the incumbent candidate, and the rule Y that elects the plurality winner between the candidates that are preferred by at least one voter to the incumbent or, if there are no such candidates, the STV winner. Now X is polynomial to manipulate as it ignores the votes. Similarly, Y is polynomial to manipulate since the manipulators should always put the candidate that they wish to win in first place, and the incumbent anywhere else in their vote. If all other voters prefer the incumbent to any other candidate, then this vote will ensure that the manipulators' preferred candidate wins. On the other hand, if the other voters prefer one ore more candidates to the incumbent, then this is the best vote for ensuring the manipulators' preferred candidate is the plurality winner. Now XTHENY is NP-hard to manipulate. We adapt the reduction used in [2] to prove that STV is NP-hard to manipulate by a single manipulator. We simply introduce an additional candidate, the incumbent into the voting profile used in this proof.

3-8 Identical examples to the weighted case.

♡

5.4 Unweighted votes, specific rules

With unweighted votes, we already know that a number of specific multi-stage voting rules are NP-hard to manipulate including STV [2], Nanson's, Baldwin's [15] and Coombs rules [10] (all with a single manipulator). We can add to this list Black's procedure. Like Borda voting on which it is based, a single manipulator can compute a manipulation of Black's procedure in polynomial time, but coordinating two manipulators makes the problem NP-hard.

Proposition 10. *Manipulation of Black's procedure with unweighted votes and two manipulators is NP-hard.*

Proof: We adapt the reduction used in the proof of Theorem 3.1 in [3] for the NP-hardness of manipulating Borda voting. This reduction is from a special case of numerical matching with target sums. It constructs an election with 5 votes, 3 fixed votes and 2 votes of the manipulators over the candidates 1 to m. We now add 6 sets of cyclic votes: $(1, 2, \ldots, m-1, m)$, $(2, 3 \ldots, m, 1)$, $\ldots$, $(m-1, m, \ldots, m-3, m-2)$, $(m, 1, \ldots, m-2, m-1)$. This has no impact on the differences in the scores between the candidates. However, it creates a Condorcet cycle so that there cannot be a Condorcet winner whatever the manipulators do with their two votes. Hence, we must pass to the second round where the winner is decided by the Borda rule. As in the proof of Theorem 3.1 in [17], there is a manipulation that makes a chosen candidate the Borda winner iff there is a solution to the numerical matching problem with target sums. Thus, computing a manipulation of $Condorcet$THEN$Borda$, which is Black's procedure, is NP-hard. ♡

Proposition 11. *Deciding whether one manipulator can make a candidate win for Black's procedure with unweighted votes is polynomial.*

Proof: We consider several cases.

Suppose no Condorcet winner exists in the profile P of votes of the non-manipulators, but there are $a \neq p$ and $b \neq p$ such that $beat_P(a, b) = beat_P(b, a)$, where $beat_P(v_1, v_2)$ is the number of times v_1 beats v_2 in P. In this case, p loses regardless of how the manipulator votes as the manipulator's vote must give an advantage of one vote to a or b. Hence, one of a or b must be the Condorcet winner.

Suppose no Condorcet winner exists in P and there is **no** $a \neq p$ and $b \neq p$ such that $beat_P(a, b) = beat_P(b, a)$. Then the manipulator casts a vote using to the greedy rule. This vote does not create a Condorcet winner that is different from p, hence it is optimum for both the Condorcet criterion and Borda rule.

Suppose there is a Condorcet winner in P, $a \neq p$. If there is no b such that $beat_P(a, b) = beat_P(b, a) + 1$ then a is the winner regardless of the manipulator's vote. Therefore, suppose there exists a set B such that $beat_P(a, b) = beat_P(b, a) + 1$, $b \in B$. If there exists b such that $score_P(a) \geq score_P(b)$ then a will be ranked below b in the manipulator's vote that is constructed based on the greedy algorithm (or we can swap a and b if their scores are equal). Therefore,we assume that $score_P(a) < score_P(b)$. Let b^* be the candidate with the minimum score $score_P$, so that $b^* = argmin_{b \in B}(score_P(b))$. The manipulator must rank a below b^* to prevent a from being the Condorcet winner. This is equivalent to assuming that $score_P(a) = score_P(b^*)$ and using the greedy algorithm to construct the manipulator's vote. If this is a successful manipulation then we are done. If it is not then there is no way to construct a successful manipulation. ♡

6 Multiple ballots

So far, we have assumed that voters vote only once. However, the THEN combinator is naturally sequential. We can therefore consider the case where voters are allowed to re-vote in each round. For example, in the French presidential elections, voters re-vote in the second stage. Such re-voting increases the potential for manipulation in two ways. First, as we illustrate here, there are elections which can only be manipulated when the manipulators vote differently in the two rounds. Of course, all those elections where manipulators can change the result by strategically voting the same way in both rounds remain manipulable. Second, as we also argue in the next section, the first round of voting reveals voters' preferences, thereby enabling manipulations to take place that require such knowledge. Third,

voters can vote strategically in the first round to give their preferred candidate an easier contest in the second round.

If voters re-vote between rounds, we add "with re-voting" to its name. Hence, plurality with runoff and re-voting is the two stage election rule used in French presidential elections in which, unless there a majority in the first round, plurality is used in the first round to select two candidates to go through to the runoff, and voters then re-vote in the second round to decide the winner of the runoff. The following example demonstrates that there exist elections where strategic voting with plurality with runoff is not possible, but is with plurality with runoff and re-voting.

Example 1. *Suppose we have 2 votes for* (a, b, p)*, 2 votes for* (b, a, p)*, 1 vote for* (b, p, a)*, 2 votes for* (p, a, b) *and 2 manipulators whose preferences are* (p, a, b)*. In addition, we suppose in the event of a tie in the first round between all 3 candidates, the manipulators' preferred candidate* p *and* a *go through to the runoff. Note that if the manipulators vote truthfully, then* p *and* b *have the most votes in the first round, and* b *wins the runoff by 5-4. To make* p *the winner, the manipulators need* a *and* p *to be in the runoff. This is possible if and only if one of the manipulators votes for* a *and the other votes for* p *in the first round. We then have a 3-way tie and, according to the tie-breaking rule,* a *and* p *go through to the runoff. If the manipulators do not re-vote in the runoff,* a *wins the runoff by 5-4. On the other hand, if the manipulators can re-vote in the runoff, both can vote for* p*, and* p *will beat* a *by 5-4.*

7 Revealed preferences

One of the strong assumptions made in much work on (the complexity of) manipulation is that the manipulators know the other voters' preferences [9]. There are many situations where this is unrealistic. When we have re-voting, it is reasonable to suppose voters' preferences have been (partially) revealed by the first round of voting. This introduces new opportunities for manipulation. Consider Black's procedure with re-voting and a manipulator who lacks any knowledge of the other voters' preferences, so votes truthfully in the first round. The following example demonstrates that this manipulator can vote strategically in the second round based on the votes revealed in the first round.

Example 2. *Suppose the first round reveals that there are 2 votes for* (a, b, p)*, 2 votes for* (b, p, a)*, 1 vote for* (p, a, b)*, and a single manipulator's truthful vote for* (p, b, a)*. There is no Condorcet winner so all candidates go through to the second round. Without re-voting,* b *has the highest Borda score in the second round and is the overall winner. On the other hand, suppose the manipulator changes their vote in the second round to* (p, a, b) *based on the preferences revealed in the first round. Then, assuming the other votes remain the same, the Borda scores of all candidates are equal. If such a 3-way tie is broken in favour of the manipulator, then the manipulator's preferred candidate* p *now wins.*

It is natural to consider more game theoretic behaviours in such two stage voting rules. Re-voting can be viewed as a finite repeated sequential game so we can use concepts like subgame perfect Nash equilibrium and backward induction to predict how agents will play strategically in each round. An interesting open question is the computational complexity of computing such strategic behaviour. This sort of strategic voting has already received some attention in the literature. For example, Bag, Sabourian and Winter prove that a class of voting rules which use repeated ballots and eliminate one candidate in each round are Condorcet consistent [1]. They illustrate this class with the *weakest link* rule in which the candidate with the fewest ballots in each round is eliminated.

It is also natural to consider iterated voting in multiple stage voting rules. After each round of voting, we might suppose that agents change their vote according to a best response

strategy, starting perhaps from a truthful vote. We can also consider the situation where the full preferences of the agents are revealed in each round of voting, as well as the situation where only partially information is revealed like total Borda scores. However, unlike previous studies like [14], candidates are also eliminated in each round.

8 Related work

As noted earlier, a number of well known voting rules like Black's procedure and Plurality with runoff are instances of this voting schema. However, there exist many other related voting rules. For example, Conitzer and Sandholm [5] studied the impact on the computational complexity of manipulation of adding an initial round of the Cup rule to a voting rule X. This initial round eliminates half the candidates and makes manipulation NP-hard to compute for several voting rule including plurality and Borda. Consider the multi-winner voting rule, *Bisect* which runs an election between given pairs of candidates, and returns the winning half of the candidates. Then Conitzer and Sandholm's study can be viewed as of the two stage voting rule $Bisect$THENX. Elkind and Lipmaa [11] extended this idea to a general technique for combining two voting rules. The first voting rule is run for some number of rounds to eliminate some of the candidates, before the second voting rule is applied to the candidates that remain. They proved that many such combinations of voting rules are NP-hard to manipulate.

Beside STV, Nanson's, Baldwin's and Coombs rule, a number of other recursively defined rules have been put forwards in the literature. For example, Tideman proposed the Alternative Smith rule [18]. This is recursively defined as $SmithSet$THEN($PluralityLoser$THEN$AlternativeSmith$). Other complex multi-stage rules have also been proposed. For example, [13] has proposed a complex rule that computes the Schwartz choice set, then iteratively applies Copeland's procedure to this set until a fixed point is reached. If several candidates remain at this point, the rule then selects the plurality winners. If there are several such winners, the rule then chooses among then according to the number of second place votes, and so on. If this still does not select a winner, a lottery is then used amongst the candidates that remain.

We recently proposed a combinator for taking the consensus of two (or more) voting rules. Given two voting rules X and Y, the combinator $X + Y$ computes the winners of X and Y and then recursively applies $X + Y$ to this set. If X and Y are majority consistent (that is, given an election with just two candidates, they both return the majority winner) then $X + Y$ is (XORY)THEN$Majority$ where XORY returns the union of the winners of X and Y.

9 Conclusions

We have considered voting rules which have multiple stages. For example, Black's procedure selects the Condorcet winner if they exist, otherwise in the second stage, it selects the Borda winner. We denoted this as $Condorcet$THEN$Borda$. Combining voting rules together in this way can increase their resistance to manipulation. For example, whilst Plurality is polynomial to manipulate with weighted votes, $Condorcet$THEN$Plurality$ is NP-hard with 3 or more candidates and a coalition of manipulators. A combination of voting rules can also inherit computational resistance to manipulation from its part. For example, we proved that computing a manipulation of Black's procedure, which is $Condorcet$THEN$Borda$, is NP-hard with weighted or unweighted votes. There are many directions for future work. For instance, it would also be interesting to consider the impact of such two stage voting on other types of control, on bribery and on issues like the computation of possible winners.

Acknowledgements

NICTA is funded by the Department of Broadband, Communications and the Digital Economy, and the Australian Research Council. The authors are also supported by the Asian Office of Aerospace Research and Development (AOARD).

References

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

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

[3] N. Betzler, R. Niedermeier, and G.J. Woeginger. Unweighted coalitional manipulation under the Borda rule is NP-hard. In T. Walsh, editor, *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*. International Joint Conference on Artificial Intelligence, 2011.

[4] F. Brandt, M. Brill, E. Hemaspaandra, and L.A. Hemaspaandra. Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates. In M. Fox and D. Poole, editors, *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010)*. AAAI Press, 2010.

[5] V. Conitzer and T. Sandholm. Universal voting protocol tweaks to make manipulation hard. In *Proceedings of 18th IJCAI*, pages 781–788. International Joint Conference on Artificial Intelligence, 2003.

[6] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate. *Journal of the Association for Computing Machinery*, 54, 2007.

[7] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate. *Journal of the Association for Computing Machinery*, 54, 2007.

[8] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate. *Journal of the Association for Computing Machinery*, 54, 2007.

[9] V. Conitzer, T. Walsh, and L. Xia. Dominating manipulations in voting wih partial information. In W. Burgard and D. Roth, editors, *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011)*. AAAI Press, 2011.

[10] J. Davies, N. Narodytska, and T. Walsh. Eliminating the weakest link: Making manipulation intractable? In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI 2012)*. AAAI Press, 2012.

[11] E. Elkind and H. Lipmaa. Hybrid voting protocols and hardness of manipulation. In *Proceedings of the 16th Annual International Symposium on Algorithms and Computation (ISAAC'05)*, 2005.

[12] P.C. Fishburn. Condorcet social choice functions. *SIAM Journal on Applied Mathematics*, 33(3):469–489, 1977.

[13] C.G. Hoag and G.H. Hallett. *Proportional Representation*. Macmillan, 1926.

[14] O. Lev and J.S. Rosenschein. Convergence of iterative voting. In *The Eleventh International Joint Conference on utonomous Agents and Multiagent Systems (AAMAS 2012)*, Valencia, Spain, June 2012.

[15] N. Narodytska, T. Walsh, and L. Xia. Manipulation of Nanson's and Baldwin's rules. In W. Burgard and D. Roth, editors, *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011)*. AAAI Press, 2011.

[16] N. Narodytska, T. Walsh, and L. Xia. Combining voting rules together. In Luc de Raedt, editor, *Proc. of the 20th European Conference on Artificial Intelligence (ECAI-2012)*, Frontiers in Artificial Intelligence and Applications. IOS Press, 2012.

[17] B. Reilly. Social choice in the south seas: electoral innovation and the borda count in the pacific island countries. *International Political Review*, 23(4):355–372, 2002.

[18] N. Tideman. *Collective Decisions And Voting: The Potential for Public Choice*. Ashgate, 2006.

Nina Narodytska and Toby Walsh
NICTA and UNSW
Sydney, Australia
Email: {nina.narodytska,toby.walsh}@nicta.com.au