Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

# Control Complexity of Schulze Voting

**Curtis Menton**[1] and **Preetjot Singh**[2]

[1]Dept. of Comp. Sci., University of Rochester, Rochester, NY, USA
[2]EECS, Northwestern University, Evanston, IL, USA

## Abstract

Schulze voting is a recently introduced voting system enjoying unusual popularity and a high degree of real-world use, with users including the Wikimedia foundation, several branches of the Pirate Party, and MTV. It is a Condorcet voting system that determines the winners of an election using information about paths in a graph representation of the election. We resolve the complexity of many electoral control cases for Schulze voting. We find that it falls short of the best known voting systems in terms of control resistance, demonstrating vulnerabilities of concern to some prospective users of the system.

## 1 Introduction

Voting has a long history as a tool for collaborative decision making. It has classically been used for political and organizational elections, but it has become increasingly important through its use in artificial intelligence and multiagent systems. Some applications include recommender systems [Ghosh *et al.*, 1999; Pennock *et al.*, 2000], consensus mechanisms for planning [Ephrati and Rosenschein, 1991] and search engine design [Lifantsev, 2000; Dwork *et al.*, 2001].

A persistent worry with elections is that they may be manipulated, harming their integrity and often calling into question the authenticity of the results. The seminal work of Gibbard and Satterthwaite [Gibbard, 1973; Satterthwaite, 1975] (and the later extension of this result [Duggan and Schwartz, 2000]) showed that in fact no reasonable voting system is immune to manipulation (i.e., strategic voting). Later, Bartholdi, Tovey, and Trick introduced computational complexity as a novel defense against this problem [Bartholdi *et al.*, 1989]. A flood of research has subsequently investigated the complexity of many voting systems under various manipulative action problems.

Here we study Schulze voting, a new and sophisticated voting system that has become fairly popular in recent years. We investigate its complexity under control, a class of manipulative action problems where the election chair changes the structure of the election to affect the result. We build on earlier work by Parkes and Xia [Parkes and Xia, 2012] and further characterize the worst-case behavior of Schulze voting under control. We find that it possesses a good but not exceptional number of control resistances, fewer than the best known natural systems. Schulze voting currently is known to posses the same control resistances as Copeland$^\alpha$ with $\alpha \in (0, 1)$, which was at one point the most resistant system known that uses the standard linear vote model [Faliszewski *et al.*, 2009b]. While it fails to stand among the most resistant systems, the popularity of the system increases the relevance of these results, perhaps calling into question the use of the system in contexts where these control actions are of concern.

### 1.1 Preliminaries

An *election* is a pair $(C, V)$ where $C$ is a finite set of candidates and $V$ is a finite collection of votes. The most common model is for a vote to be given as a strict linear ordering of the candidates, though other models are sometimes used as well. Most notably, approval voting uses binary vectors over the candidates as votes.

A *voting system* is a mapping from an election $(C, V)$ to a set $C' \subseteq C$ of winners. We do not require a voting system to select only a single winner: This would require us to build in tiebreaking mechanisms while they are better left as separate procedures. We do not require a voting system even to always select any winners at all, as there are electoral contexts where selecting no candidates is a reasonable outcome. For instance, the voting system used to select entrants for the Baseball Hall of Fame can select no players if none reach a certain threshold of support.

### 1.2 Schulze Voting

Schulze voting is a Condorcet voting system recently introduced by Marcus Schulze [Schulze, 2011]. It was designed in part to effectively handle candidate *cloning*: In many voting systems, the inclusion of several similar candidates ends up diluting their support and lessening the influence of their supporters. It has a somewhat more complex winner procedure than other common voting systems, requiring the use of a graph best-path finding algorithm, but it is still solvable in polynomial time, rendering Schulze a tractable voting system.

Schulze voting has garnered a high level of real-world use, especially in the free software and free culture communities. Users include several national branches of the Pirate Party, the Wikimedia foundation, many free-software projects such

as Ubuntu, Debian, Gentoo, and KDE, and even MTV. This level of real-world usage is unusual for a new, academically proposed and studied voting system, but it makes Schulze voting more compelling to analyze.

As a typical Condorcet system, the winners are determined by examining the pairwise contests between candidates. We will thus introduce some useful functions and notation. The *advantage function* is a function on pairs of candidates where $adv(a, b)$ is the number of voters in the given election that prefer $a$ to $b$ (that is, rank $a$ higher than $b$ in their preferences). The *net advantage function* gives the net difference in advantage for one candidate over another. We define the net advantage between $a$ and $b$ to be the following: $netadv(a, b) = adv(a, b) - adv(b, a)$.

It is easily possible to construct the net advantage function given a collection of votes, but it is interesting to note that we can easily convert in the other direction as well. Given a net advantage function where either all the scores are even or all the scores are odd, we can construct an equivalent set of votes in time polynomial in the number of candidates and in the maximum magnitude of any score in the net advantage function. This method is due to McGarvey [McGarvey, 1953]. We will use this method later to more easily construct elections in the course of reductions, in order to avoid having to explicitly specify the votes.

The winners in a Schulze election are determined as follows. Generate the net advantage scores for the election, and represent this data as a graph, with vertices for candidates and directed edges denoting net advantage scores. Determine the strongest paths in this graph between every pair of candidates by the following metric: The weight of a path is the lowest weight edge in the path. This is the "bottleneck" metric. We can adapt the Floyd-Warshall dynamic programming algorithm to find the weight of such paths in polynomial time [Schulze, 2011].

Once we have the best path weights, we build another graph with the same vertices where there is a directed edge from $a$ to $b$ if the best path from $a$ to $b$ is stronger than the best path from $b$ to $a$. The winners of the election are the candidates with no in-edges in this final graph. That is, they are the candidates that are not beaten by any other candidate in relative best-path strength.

Note that a Condorcet winner, in the election graph, will have positive weight edges to every other candidate, and so they will clearly have the better paths to every other candidate and be the only winner of the election.

**Example Election**

Consider a Schulze election over candidates $\{a, b, c, d\}$ with the following table expressing the net advantage function.

| | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| $a$ | | 4 | -2 | 0 |
| $b$ | -4 | | 4 | -2 |
| $c$ | 2 | -4 | | 6 |
| $d$ | 0 | 2 | -6 | |

We can also represent this election as a graph as in Figure 1. Now we must find the best paths between pairs of candidates. Note that there are several Condorcet cycles and there is no clearly dominant candidate: Each of them loses to at least one other candidate, and each candidate has a path to every other candidate. The following are the Schulze best path scores.

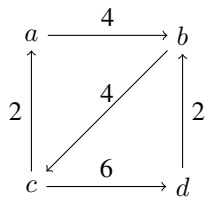


Figure 1: Election graph for the example Schulze election.

| | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| $a$ | | 4 | 4 | 4 |
| $b$ | 2 | | 4 | 4 |
| $c$ | 2 | 2 | | 6 |
| $d$ | 2 | 2 | 2 | |

The only winner of the election will be $a$, as it has a stronger path to every other candidate than those candidates have back to $a$.

## 1.3 Election Manipulative Actions

There are several classes of manipulative action problems studied in computational social choice. The primary classes of these problems are *manipulation*, where a voter or voters strategically vote to affect the result of an election [Bartholdi *et al.*, 1989], *bribery*, where an outside briber pays off voters to change their votes [Faliszewski *et al.*, 2009a], and *control*, where an election organizer alters the structure of an election to change the result [Bartholdi *et al.*, 1992].

These problems are formalized as decision problems as is standard in computer science and we study their complexity in various voting systems. There is some standard terminology about the behavior of voting systems under manipulative actions. A voting system is *immune* to a manipulative action if the action can never change the result of an election in the voting system, and the system is *susceptible* otherwise. If a voting system is susceptible to some action and the decision problem is in P, then it is *vulnerable* to it, and the system is *constructively vulnerable* if additionally the corresponding search problem is in P. If the decision problem is NP-hard, the voting system is *resistant* to the action. The original versions of these problems are the *constructive* cases, where the goal is to make a particular candidate win, as opposed to *destructive* cases, where the goal is to prevent a particular candidate from winning. These were introduced by [Conitzer *et al.*, 2007] in the context of manipulation and by [Hemaspaandra *et al.*, 2007] in the context of control.

Additionally, there are differing versions of these problems based on our tie-handling philosophy. The work presented here follows the nonunique-winner model, where we claim success in our manipulative action when the preferred candidate is one of possibly several winners in the election in the

constructive case, or is not among the set of winners in the election in the destructive case. In contrast, the original paper exploring control [Bartholdi *et al.*, 1992] used the unique-winner model, where the goal is to cause a candidate to be the only winner of an election, or in the destructive cases for the candidate to be not a unique winner. Both models are common in the literature.

### 1.4 Control

Control encompasses actions taken to change the structure of an election to achieve a desired result. This includes adding or deleting candidates or voters, or partitioning either candidates or voters and performing initial subelections, the survivors of which go on to a final election. The study of election control was initiated with [Bartholdi *et al.*, 1992], and many subsequent papers have investigated the complexity of control in various voting systems.

The various control problems loosely model many real-world actions. The cases of adding and deleting voters correspond to voter registration drives and voter suppression efforts. The cases of adding and deleting candidates correspond to ballot-access procedures that effectively remove many candidates from elections. Cases of partitioning voters are similar to the real-world practice of gerrymandering (though that has additional geographic constraints), and cases of partitioning candidates correspond to primary elections or runoffs.

Much of the study of control has had the goal of finding voting systems that are resistant to a large number of cases of control. [Faliszewski *et al.*, 2009b] showed that Llull and Copeland voting are resistant to every case of constructive control, [Hemaspaandra *et al.*, 2009] constructed unnatural hybrid voting systems that resist every case of control, proving that such systems can exist, and [Erdélyi and Rothe, 2010; Erdélyi and Fellows, 2010; Erdélyi *et al.*, 2011] showed that fallback voting resists 20 out of the 22 standard cases of control, and so it stands along with normalized range voting [Menton, 2012] as the most resistant natural voting system. No natural system resistant to every case of control has yet been found. We define one case of control completely below; other control cases are defined similarly. For the rest, see [Hemaspaandra *et al.*, 2007].

#### Control by Deleting Candidates

**Given** An election $E = (C, V)$, a distinguished candidate $p \in C$, and $k \in \mathbb{N}$.

**Question (Constructive)** Is it possible to make $p$ a winner of an election $(C - C', V)$ with some $C' \subseteq C$ where $\|C'\| \leq k$?

**Question (Destructive)** Is it possible to make $p$ not a winner of an election $(C - C', V)$ with some $C' \subseteq (C - \{p\})$ where $\|C'\| \leq k$?

## 2 Results

The complexity of the manipulative action problems for Schulze voting was studied in [Parkes and Xia, 2012], which proved resistance for bribery and some of the control cases, and showed that constructive manipulation is easy with a single manipulator. However several control cases still remained open and the case of constructive manipulation with multiple manipulators has just very recently been shown to have a polynomial-time algorithm [Gaspers *et al.*, 2013]. Our new results are summarized in Table 1, which shows the behavior of Schulze voting, as well as several other voting systems, under control. We include here proofs of a selection of our results.

**Theorem 2.1.** *Schulze voting is resistant to constructive control by partition of voters, ties eliminate.*

*Proof.* CC-PV-TE Case

We prove this case through a reduction from 3SAT, defined as follows [Garey and Johnson, 1979].

**Given** A set $U$ of variables and a collection $Cl$ of clauses over $U$ such that each clause $c \in Cl$ has $||c|| = 3$.

**Question** Is there a satisfying truth assignment for $Cl$?

We will reduce from a 3SAT instance $(U, Cl)$ and output a control instance $((C, V), p)$. The reduction assumes $||U||$ to be even, so we will modify the instance if necessary by adding a dummy variable. The candidate set $C$ will be as follows:

- A distinguished candidate $p$.
- A pair of candidates $c_i, c_i'$ for every clause $c_i \in Cl$.
- A pair of auxiliary candidates $c_1^\star, c_2^\star$.
- A candidate $x_i$ for every variable $x_i \in U$.
- An auxiliary candidate $a$.

We will call the ordering $c_1 > c_1' > c_2 > c_2' > \dots$ the *standard ordering* of the clause candidates. The candidates $c_1^\star$ and $c_2^\star$ will be *clones* of each other, always appearing next to each other in every vote, and each will be set to be above the other in exactly half of all the votes.

The voter set $V$ will consist of the following groups.

1. For each variable $x_i$, where $D$ is the set of clauses satisfied by $x_i$ assigned to true, a voter:
   - ranking $x_i$ first,
   - ranking $c_1^\star, c_2^\star$ next,
   - ranking the clause candidates in reverse standard order next, except with the pair of candidates $c_j, c_j'$ flipped for every $c_j \in D$,
   - ranking $p$ next,
   - ranking the candidates in $U - \{x_i\}$ next, and
   - ranking $a$ last.
2. For each variable $x_i$, a voter similar to those in the previous group except with $D$ being the set of clauses satisfied with $x_i$ assigned to false.
3. $||U|| - 3$ voters preferring $a$ over every $x_i$ over $p$ over the clause candidates in standard order over $c_1^\star$ and $c_2^\star$.
4. 1 voter preferring $a$ over $p$ over every $x_i$ over the clause candidates in standard order over $c_1^\star$ and $c_2^\star$.
5. 1 voter preferring $a$ over $p$ over the clause candidates in standard order over $c_1^\star$ and $c_2^\star$ over every $x_i$.
6. 1 voter preferring every $x_i$ over $p$ over $a$ over the clause candidates in standard order over $c_1^\star$ and $c_2^\star$.

| Control by | Tie Model | Plurality | | Approval | | Fallback | | Schulze | |
|---|---|---|---|---|---|---|---|---|---|
| | | C | D | C | D | C | D | C | D |
| Adding Candidates | | R | R | I | V | R | R | R | S |
| Adding Candidates (unlimited) | | R | R | I | V | R | R | **R** | S |
| Deleting Candidates | | R | R | V | I | R | R | **R** | S |
| Partition of Candidates | TE | R | R | V | I | R | R | **R** | **V** |
| | TP | R | R | I | I | R | R | **R** | **V** |
| Run-off Partition of Candidates | TE | R | R | V | I | R | R | **R** | **V** |
| | TP | R | R | I | I | R | R | **R** | **V** |
| Adding Voters | | V | V | R | V | R | V | R | R |
| Deleting Voters | | V | V | R | V | R | V | R | R |
| Partition of Voters | TE | V | V | R | V | R | R | **R** | **R** |
| | TP | R | R | R | V | R | R | **R** | **R** |

Table 1: Control behavior under Schulze voting and other voting systems for comparison. V, R, S, and I stand for vulnerable, resistant, susceptible, and immune. Results proved in this paper in bold, other results from [Erdélyi and Rothe, 2010; Erdélyi and Fellows, 2010; Erdélyi *et al.*, 2011; Hemaspaandra *et al.*, 2007; Parkes and Xia, 2012].

7. 1 voter preferring every $x_i$ over $a$ over $p$ over the clause candidates in standard order over $c_1^\star$ and $c_2^\star$.
8. 1 voter preferring $c_1^\star$ and $c_2^\star$ over the clause candidates in reverse standard order over $p$, except flipping adjacent pairs of candidates, starting by putting the first clause candidate over $c_1^\star$ and $c_2^\star$, and ending by putting $p$ over the last clause candidate, following this sequence with $a$ over every $x_i$.

The unspecified relative rankings of the variable candidates will be set according to the McGarvey method [McGarvey, 1953] to equalize the contests between variable candidates within any "valid assignment" subset of the variable assignment voters, and within the other groups of voters. We assume that $||U||$ is even so that we can evenly equalize these contests.

If we are mapping from a positive 3SAT instance, it will be possible to make $p$ win the final election through this type of control. The partition will be as follows: The first partition will contain all of the third through eighth groups of voters, and will contain voters from the first two groups corresponding to a satisfying assignment. The result of adding the voters from the third through eighth groups will be to give $p$ a path of strength $||U||$ through all the clause candidates, with some edges of strength $||U|| + 2$, to give each $x_i$ $||U|| - 4$ votes over $p$, and to give $a$ $||U|| - 2$ votes over $p$. By adding the variable assignment voters corresponding to the satisfying assignment, we decrease the weaker edges in the clause candidate path by by no more than $||U|| - 2$, increase every $p$-$x_i$ edge by $||U|| - 2$, and increase the $p$-$a$ edge by $||U||$. The result is that we boost $p$ over $a$ and over the variable candidates while preserving the path from $p$ through the clause candidates. Thus $p$ will beat every candidate and win their subelection. In the other subelection, with the remaining variable assignment candidates, $c_1^\star$ and $c_2^\star$ will be winners, and with two or more winners no candidates will be promoted. $p$ will thus be alone in the final election and will win.

If we map to a positive control instance, the mapped-from 3SAT instance will be satisfiable. No voter prefers $p$ outright, so we must balance their votes against the other candidates to have a chance. The voters in the first two groups prefer $p$ over $a$ and they mostly prefer $p$ over the variable candidates, but they prefer the clause candidates over $p$. The voters in the third through eighth groups can boost $p$ over the clause candidates, but they give $a$ and the variable candidates an advantage over $p$. To make $p$ the only winner of a subelection, we have to carefully select the voters from the first two groups to keep $p$ ahead of the variable candidates while preserving $p$'s path through the clause candidates. To do so we must ensure that at least one voter prefers $c_i$ over $c_i'$ for every clause $c_i$, forcing the corresponding assignment to satisfy every clause, and that only one voter prefers each $x_i$ over $p$, forcing the corresponding assignment to assign each variable as either "true" or "false" but not both. These voters thus correspond to a satisfying assignment, so the 3SAT instance must be satisfiable. □

**Theorem 2.2.** *Schulze voting is resistant to constructive control by deleting candidates.*

*Proof.* CC-DC Case

We prove this case through a reduction from 3SAT. Given a 3SAT instance $(U, Cl)$ we will construct a control instance $((C, V), p, k)$ as follows. Our deletion limit $k$ will be equal to $||Cl||$. The candidate set will be the following:

- Our special distinguished candidate $p$.
- For each clause $c_i \in Cl$, $k+1$ candidates $c_i^1, \ldots c_i^{k+1}$.
- A candidate for each literal $x_i^j$ in each of the clauses (where $x_i^j$ is the $j$th literal in the $i$th clause).
- For each pair of literals $x_i^j, x_m^n$ where one is the negation of the other, $k+1$ candidates $n_{i,j,m,n}^1, \ldots n_{i,j,m,n}^{k+1}$.
- An additional auxiliary candidate $a$.

The $k+1$ candidates for each clause and for each conflicting pair of literals are treated as copies of each other, and are included to prevent successful control by simply deleting the tough opponents rather than solving the more difficult problem corresponding to the 3SAT instance.

The voters will be constructed according to the McGarvey method [McGarvey, 1953] such that we have the following relationships between candidates.

- For the three literal candidates $x_i^1, x_i^2, x_i^3$ in a clause $c_i$, $c_i^j$ beats $x_i^1$ (for all $j$), $x_i^1$ beats $x_i^2$, $x_i^2$ beats $x_i^3$, and $x_i^3$ beats $p$, all by two votes.
- For a pair of literal candidates $x_i^j$ and $x_m^n$ that are negations of each other, each beats $n_{i,j,m,n}^l$ (for all $l$) by two votes.
- Every negation candidate $n_{i,j,m,n}^l$ beats $p$ by two votes.
- $p$ beats $a$ by two votes.
- $a$ beats every $x_i^j$ by two votes.

This completes the specification of the reduction, which clearly can be performed in polynomial time. The intuition is as follows: Deleting a literal corresponds to assigning that literal to be true, and to make $p$ win we must "assign" literals that satisfy every clause without ever "assigning" a variable to be both true and false, so a successful deletion corresponds to a valid satisfying assignment.

We will now show that if $(U, Cl)$ is a positive 3SAT instance, $p$ can be made a winner of this election by deleting $k$ candidates. We will delete one literal candidate for each clause, selecting a literal that is satisfied by the satisfying assignment for $Cl$. This will require us to delete $||Cl||$ candidates, which is equal to our deletion limit $k$. By deleting these candidates, we break all the paths from the clause candidates to $p$, so now $p$ is tied with each clause candidate instead of being beaten by them. Also, since we deleted literals that were satisfied according to a satisfying assignment, we must not have deleted any pair of literals that were the negations of each other, so we will still have a path from $p$ to each negation candidate. Thus $p$ at least ties every other candidate in Schulze score, so they will be a winner.

If we map to a positive control instance, our 3SAT instance must be positive as well. First, the most serious rivals to $p$ are the clause candidates, who each have a path of strength two to $p$ while $p$ only has a path of strength 0 to them. Since there are many duplicates of each of them, we cannot remove them directly but must instead remove other vertices along the paths to $p$ to remove the threat. The deletion limit allows us to delete one candidate for every clause. However, we must choose which ones to delete carefully. If we delete a literal and a different one that is the negation of the first, we destroy our only paths to the corresponding negation candidate. Thus we must delete one literal for each clause, while avoiding deleting a variable and its negation. If there is a successful way to do this, there will be a satisfying assignment for the input instance that we can generate using our selection of deleted/satisfied literals (arbitrarily assigning variables that were not covered in this selection). □

**Theorem 2.3.** *Schulze voting is vulnerable to destructive control by partition of candidates and destructive control by runoff-partition of candidates in either the ties-promote or ties-eliminate model.*

Schulze voting is vulnerable to each of the variants of destructive control by partition of candidates. We will show this, and additionally show that these problems are easy for a broad class of Condorcet voting systems.

In the nonunique-winner model that we are following here, both of the pairs destructive control by partition of candidates, ties eliminate and destructive control by runoff partition of candidates, ties eliminate; and destructive control by partition of candidates, ties promote and destructive control by runoff partition of candidates, ties promote are in fact the same problems, that is, the same sets. This fact was discovered in [Hemaspaandra *et al.*, 2012] by noting shared alternative characterizations of these problems. The nominal difference is that in the "partition of candidates" case, the candidate set is partitioned and only one part undergoes a culling subelection while the other part gets a bye, while in the "runoff partition of candidates" case, both parts of the partition first face a subelection. In truth they are much simpler problems, and identical (within the same tie-handling model). Namely, the sets DC-PC-TE and DC-RPC-TE are equivalent to the set of instances $((C, V), p)$ where there is some set $C' \subseteq C$ with $p \in C'$ such that $p$ is not a unique winner of the election $(C', V)$. The sets DC-PC-TP and DC-RPC-TP are equivalent to the set of instances $((C, V), p)$ where there is some set $C' \subseteq C$ with $p \in C'$ such that $p$ is not a winner of the election $(C', V)$. We will build algorithms for these cases aided by these characterizations.

These algorithms are optimal in a class of voting systems that are Condorcet voting systems that also possess a weaker version of the Condorcet criteria—where if there are any candidates that do no worse than tying in pairwise contests with other candidates (known as weak Condorcet winners), they will be winners (possibly, but not necessarily unique) of the election. There can potentially be one or more than one such candidates. Schulze voting is such a voting system: If a candidate is a weak Condorcet winner, doing no worse than tying against any other candidate, no candidate can have a path to that candidate of strength greater than 0, and since that candidate at least ties every other candidate, he or she has a path of strength at least 0 to every other candidate. Thus he or she is unbeaten in Schulze score, and will be a winner.

Other Condorcet voting systems that possess this property include Copeland[1], Minimax, and Ranked Pairs.

*DC-PC-TP/DC-RPC-TP Case.* Recall our alternate characterization for these problems, that the set of positive instances is the same as the set of instances $((C, V), p)$ where there is some set $C' \subseteq C$ with $p \in C'$ such that $p$ is not a winner of the election $(C', V)$. Thus, finding if we have a positive instance is very similar to the problem of finding if we have a positive instance in the destructive control by deleting candidates problem, except that there is no longer a limit on the number of candidates we can delete. Thus we do not have to carefully limit the number of deleted candidates, but we can freely delete as many candidates as necessary to put $p$ in a losing scenario.

Given a control instance $((C, V), p)$, all we must do is see if there is any candidate $a \in C$, $a \neq p$, such that $netadv(a, p) > 0$. If such a candidate exists, we let our first partition be $\{a, p\}$ and let the second be $C - \{a, p\}$. $p$ will lose their initial election to $a$, and be eliminated from the election.

If there is no such candidate, $p$ is a weak Condorcet winner, and thus they will always be a winner of the election among any subset of the candidates, as they will be a weak Condorcet winner (at least, or possible a Condorcet winner) among any subset of the candidates. Thus our algorithm will indicate failure. We can perform this check through a simple examination of the net advantage scores which can be generated from an election in polynomial time, and so this algorithm runs in polynomial time. Thus Schulze voting (and any other system meeting the aforementioned criteria) is vulnerable and constructively vulnerable to DC-PC-TP/DC-RPC-TP. □

*DC-PC-TE/DC-RPC-TE Case.* These cases have the alternate characterization that the set of positive instances is equivalent to the set of instances $((C, V), p)$ where there is some set $C' \subseteq C$ with $p \in C'$ such that $p$ is not a unique winner of the election $(C', V)$, that is, there are multiple winners, or no winners, or there is a single winner that is not $p$. This case thus differs only slightly from the DC-PC-TP/DC-RPC-TP case and we can create a very similar simple algorithm.

Given a control instance $((C, V), p)$, we can find a successful action if one exists by simply checking if there is any candidate $a \in C$, $a \neq p$, such that $netadv(a, p) \geq 0$. If so, we let our first partition be $\{a, p\}$ and let the second be $C - \{a, p\}$. This results in $p$ either not being a winner of the subelection at all or being a winner along with $a$ if the net advantage score is 0. Either way, $p$ will not be promoted to the final election and thus they will not be a winner of the final election. If there is no such candidate, $p$ is a Condorcet winner, and they will be a Condorcet winner among any subset of the candidates, so this type of control will never be possible, and our algorithm will indicate that. As before, generating the net advantage function is easily possible in polynomial time, and the simple check will only take polynomial time, so Schulze voting (and any other system meeting the aforementioned criteria) is vulnerable and constructively vulnerable to DC-PC-TE/DC-RPC-TE. □

**Lemma 2.4.** *Each of destructive control by adding candidates, destructive control by unlimited adding of candidates, and destructive control by deleting candidates in Schulze voting polynomial-time Turing reduces to* path-preserving vertex cut.

*Proof.* We define path-preserving vertex cut as follows.

**Given** A directed graph $G = (V, E)$, distinct vertices $s, t \in V$, and a deleting limit $k \in \mathbb{N}$.

**Question** Is there a set of vertices $V'$, $||V'|| \leq k$, such that the induced graph on $G$ with $V'$ removed contains a path from $t$ to $s$ but not any path from $s$ to $t$?

We note that the standard vertex cut problem is well-known to be solvable in polynomial time, but to the best of our knowledge the complexity of this variant is unknown (other than that it is clearly in NP). Given this result, Schulze voting will be vulnerable to each of these control cases if this problem is found to be in P.

We will describe how these control cases can reduce to this problem, first handling the DC-DC case. Our input will be a DC-DC instance $((C, V), p, k)$. Since this destructive case of control, we only have to make the distinguished candidate $p$ not a winner rather than making any particular candidate positively a winner. To do this we must alter the election so that some other candidate has a better path to $p$ than $p$ has to that candidate. Since this is a case of control by deleting candidates, we can only eliminate paths in the election graph, not create them. So we have to try to succeed by breaking the paths from $p$ to some other candidate.

We can first handle the cases where $p$ is a Condorcet winner (control will be impossible) or where $p$ is already beaten by some candidate (control is trivial). Otherwise, we will loop through the candidates and see if we can make any of them beat $p$.

For each candidate $a$, we will first find the subgraph containing all maximum-strength paths to $p$ (using a modified Floyd-Warshall algorithm), and also the subgraph containing all paths from $p$ to $a$ at least as strong as the strongest $a$-$p$ path. We will then take the union of these graphs, disregard the edge weights, and apply a subroutine for path-preserving vertex cut. This will tell us if there is any way to cut all of the strong paths from $p$ to $a$ while preserving one of the strongest ones from $a$ to $p$. If we ever get a positive result from this call, we can indicate success. Now, it may be that we can succeed by preserving a path other than the strongest path from $a$ to $p$. So if we failed, we repeat the process except considering paths from $a$ to $p$ and from $p$ to $a$ that are a little less strong, going down as far as the strength of the $p$-$a$ edge (which is not cuttable). This loop is polynomially bounded in the number of edges, as there can only be as many path strengths as there are edges. We indicate failure if we never achieve success with any vertex or path strength.

The other cases can be handled similarly. The difference is that they are adding-candidates problems instead of deleting-candidates problems, but we can reduce them to modified deleting-candidates problems by including all of the addable vertices and then solving them as deleting-candidates problems where we can only delete the vertices in the added sets. Additionally, in the non-unlimited case, we would be restricted as to how many of the vertices are added, so we would have to only look at paths from $a$ to $p$ that use a limited number of the added vertices. □

## 3 Conclusions

We found that Schulze voting, despite its promisingly complicated winner determination method, does not possess resistance to every case of control. These results may be of interest to some of the system's many users, especially those concerned with computational resistance to control. Our unresolved control cases are of course of interest, and they may be resolved through development of a polynomial-time algorithm for the path-preserving vertex cut problem.

## 4 Acknowledgements

The authors thank Dr. Edith Hemaspaandra, Dr. Lane A. Hemaspaandra and Dr. Muthu Venkitasubramaniam for their guidance. The first author has been supported in part by grants NSF-CCF-0915792 and NSF-CCF-1101479.

## References


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

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

[Conitzer *et al.*, 2007] V. Conitzer, T. Sandholm, and J. Lang. When are elections with few candidates hard to manipulate? *Journal of the ACM*, 54(3):1–33, 2007.

[Duggan and Schwartz, 2000] J. Duggan and T. Schwartz. Strategic manipulability without resoluteness or shared beliefs: Gibbard–Satterthwaite generalized. *Social Choice and Welfare*, 17(1):85–93, 2000.

[Dwork *et al.*, 2001] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International World Wide Web Conference*, pages 613–622. ACM Press, March 2001.

[Ephrati and Rosenschein, 1991] E. Ephrati and J. Rosenschein. The Clarke Tax as a consensus mechanism among automated agents. In *Proceedings of the 9th National Conference on Artificial Intelligence*, pages 173–178. AAAI Press, July 1991.

[Erdélyi and Fellows, 2010] G. Erdélyi and M. Fellows. Parameterized control complexity in bucklin voting and in fallback voting. In *Proceedings of the 3rd International Workshop on Computational Social Choice*, pages 163–174, 2010.

[Erdélyi and Rothe, 2010] G. Erdélyi and J. Rothe. Control complexity in fallback voting. In *Proceedings of the 16th Australasian Theory Symposium*, pages 39–48, January 2010.

[Erdélyi *et al.*, 2011] G. Erdélyi, L. Piras, and J. Rothe. The complexity of voter partition in Bucklin and fallback voting: Solving three open problems. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 837–844, May 2011.

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

[Faliszewski *et al.*, 2009b] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence*, 35:275–341, 2009.

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

[Gaspers *et al.*, 2013] S. Gaspers, T. Kalinowski, N. Narodytska, and T. Walsh. Coalitional manipulation for schulze's rule. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[Ghosh *et al.*, 1999] S. Ghosh, M. Mundhe, K. Hernandez, and S. Sen. Voting for movies: The anatomy of recommender systems. In *Proceedings of the 3rd Annual Conference on Autonomous Agents*, pages 434–435. ACM Press, 1999.

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

[Hemaspaandra *et al.*, 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.

[Hemaspaandra *et al.*, 2009] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Hybrid elections broaden complexity-theoretic resistance to control. *Mathematical Logic Quarterly*, 55(4):397–424, 2009.

[Hemaspaandra *et al.*, 2012] E. Hemaspaandra, L. Hemaspaandra, and C. Menton. Search versus decision for election manipulation problems. Technical Report TR-971, Department of Computer Science, University of Rochester, 2012.

[Lifantsev, 2000] M. Lifantsev. Voting model for ranking web pages. In *Proceedings of the International Conference on Internet Computing*, pages 143–148, 2000.

[McGarvey, 1953] D. McGarvey. A theorem on the construction of voting paradoxes. *Econometrica*, 21(4):608–610, 1953.

[Menton, 2012] C. Menton. Normalized range voting broadly resists control. *Theory of Computing Systems*, December 2012.

[Parkes and Xia, 2012] D. Parkes and L. Xia. A complexity-of-strategic-behavior comparison between Schulze's rule and ranked pairs. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pages 1429–1435, 2012.

[Pennock *et al.*, 2000] D. Pennock, E. Horvitz, and C. Giles. Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 729–734. AAAI Press, July/August 2000.

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

[Schulze, 2011] M. Schulze. A new monotonic, clone-independent, reversal symmetric, and Condorcet-consistent single-winner election method. *Social Choice and Welfare*, 36:267–303, 2011.