Stackelberg Voting Games: Computational Aspects and Paradoxes*

Lirong Xia
Department of Computer Science
Duke University
Durham, NC 27708, USA
lxia@cs.duke.edu

Vincent Conitzer
Department of Computer Science
Duke University
Durham, NC 27708, USA
conitzer@cs.duke.edu

Abstract

We consider settings in which voters vote in sequence, each voter knows the votes of the earlier voters and the preferences of the later voters, and voters are strategic. This can be modeled as an extensive-form game of perfect information, which we call a *Stackelberg voting game*.

We first propose a dynamic-programming algorithm for finding the backward-induction outcome for any Stackelberg voting game when the rule is anonymous; this algorithm is efficient if the number of alternatives is no more than a constant. We show how to use *compilation functions* to further reduce the time and space requirements.

Our main theoretical results are paradoxes for the backward-induction outcomes of Stackelberg voting games. We show that for any $n \geq 5$ and any voting rule that satisfies non-imposition and with a low *domination index*, there exists a profile consisting of n voters, such that the backward-induction outcome is ranked somewhere in the bottom two positions in almost every voter's preferences. Moreover, this outcome loses all but one of its pairwise elections. Furthermore, we show that many common voting rules have a very low ($= 1$) domination index, including all majority-consistent voting rules. For the plurality and nomination rules, we show even stronger paradoxes.

Finally, using our dynamic-programming algorithm, we run simulations to compare the backward-induction outcome of the Stackelberg voting game to the winner when voters vote truthfully, for the plurality and veto rules. Surprisingly, our experimental results suggest that on average, more voters prefer the backward-induction outcome.

Introduction

Voting is a useful methodology that allows multiple agents to aggregate their preferences over alternatives and make a group decision. In the standard voting setting, each voter (agent) reports a linear order over (strict ranking of) the alternatives; moreover, the voters are generally assumed to do so *simultaneously* (or without knowledge of previously reported votes, which is equivalent from a game-theoretic perspective). Then, a voting rule is applied to the profile (vector) of reported linear orders, producing a winning alternative.

An important concern is that voters may vote strategically rather than truthfully. That is, a voter may report a *false* vote that does not represent her true preferences to make herself better off. This phenomenon is called *manipulation*; if the voting rule r is such that no voter can ever benefit from manipulating, then r is said to be *strategy-proof*. Unfortunately, there are some very minimal conditions that are not satisfied by any strategy-proof voting rule, according to the celebrated Gibbard-Satterthwaite theorem (Gibbard 1973; Satterthwaite 1975).

This raises the following fundamental question: if voters vote strategically, what outcome can we expect? It is natural to turn to game theoretic solution concepts to answer this question. One approach is to consider the game where all voters vote at the same time, and study the equilibria of this game. Unfortunately, even in a complete-information setting where all voters' preferences are common knowledge, this leads to an extremely large number of equilibria, many of them bizarre. For example, in a plurality election (where everyone votes for a single alternative), it may be the case that all voters prefer alternative a to both b and c. Nevertheless, in one equilibrium of this game, all voters will vote for either b or c. This equilibrium is quite robust, because voting for a is a waste, given that nobody else is expected to vote for a. There has been some work exploring different solution concepts in simultaneous-move voting games—e.g., (Farquharson 1969; Moulin 1979)—but in some sense, the equilibrium selection issue in the above example is inherent in settings where voters vote simultaneously.

However, in many practical situations, the voters vote one after another, and the later voters know the votes cast by the earlier voters. For example, consider online systems that allow users to rate movies or other products. This is the setting that we consider in this paper. We assume that voters' preferences over the alternatives are strict; we also make a complete-information assumption that the voters' preferences are common knowledge (among the voters themselves, though not necessarily to the election organizer).[1] This re-

*We thank Edith Elkind and anonymous AAAI reviewers for helpful comments. Lirong Xia is supported by a James B. Duke Fellowship and Vincent Conitzer is supported by an Alfred P. Sloan Research Fellowship. We also thank NSF for support under award numbers IIS-0812113 and CAREER 0953756.

[1] While this is clearly a simplifying assumption, it approximates the truth in many settings, and with this assumption we do not need

sults in an extensive-form game of perfect information that can be solved by backward induction. In sharp contrast to the simultaneous-move setting, this results in a unique outcome (winning alternative). We refer to this game as *Stackelberg voting game*.

Related work and discussion. The idea of modeling a voting process in which voters vote one after another as an extensive-form game is not new. Sloth (1993) studied elections with two alternatives, as well as settings with more alternatives where a pairwise decision between two options is made at every stage. She relates the outcomes of this process to the multistage sophisticated outcomes of the game (McKelvey and Niemi 1978; Moulin 1979). In the extensive-form games studied by Dekel and Piccione (2000), multiple voters can vote simultaneously in each stage. They compare the equilibrium outcomes of these games to the outcomes of the *symmetric equilibria* of their simultaneous counterparts. Battaglini (2005) studies how these results are affected by the possibility of abstention and a small cost of voting.

Our approach is significantly different from the previous approaches in several aspects. First, the prior work focuses mostly on the case of two alternatives or, in the case of multiple alternatives, on particular voting procedures; in contrast, we consider general (anonymous) voting rules with any number of alternatives, and correspondingly derive very general paradoxes. Second, we also study how the backward-induction outcome can be efficiently computed, and we use these algorithmic insights in simulations to evaluate the quality of the Stackelberg voting game's outcome "on average."

Desmedt and Elkind (2010) simultaneously and independently studied a similar setting in which voters vote sequentially under the plurality rule, and showed several types of paradoxes. In their model, voters are allowed to abstain, and voting comes at a small cost. They assume random tie-breaking and therefore need to consider expected utilities. Their paradoxes are significantly different from ours.

Preliminaries

Let $\mathcal{X}$ be the set of *alternatives*, $|\mathcal{X}| = m$. A vote V is a linear order over $\mathcal{X}$. The set of all linear orders over $\mathcal{X}$ is denoted by $L(\mathcal{X})$. An *n-profile* P is a collection of n votes, that is, $P \in L(\mathcal{X})^n$. A *voting rule* r (for m alternatives and n voters) is a mapping that assigns to each profile a unique winning alternative. That is, $r : L(\mathcal{X})^n \to \mathcal{X}$. Some voting rules are listed below.

• **(Positional) scoring rules**: Given a *scoring vector* $\vec{v} = (v(1), \ldots, v(m))$, for any vote $V \in L(\mathcal{X})$ and any $c \in \mathcal{X}$, let $s(V, c) = v(j)$, where j is the rank of c in V. For any profile $P = (V_1, \ldots, V_n)$, let $s(P, c) = \sum_{i=1}^{n} s(V_i, c)$. The rule will select $c \in \mathcal{X}$ so that $s(P, c)$ is maximized. Some examples of positional scoring rules are *plurality*, for which the scoring vector is $(1, 0, \ldots, 0)$; and *veto*, for which the scoring vector is $(1, \ldots, 1, 0)$.

• **Nomination:** Nom is defined as follows. For any profile P, we let $Tops(P)$ denote the set of alternatives that are ranked in the top position in at least one vote in P (the alternatives that have been "nominated"). We choose the first alternative in $Tops(P)$ according to a fixed order. That is, $\text{Nom}(P) = c_{i^*}$ where $i^* = \min\{i : c_i \in Tops(P)\}$.

There are also certain *criteria* that voting rules can satisfy. We now give two criteria and some example rules that satisfy them. (We do not define these example rules here because they are well known in the computational social choice community, and a definition is not technically necessary because the results in this paper will hold for all rules satisfying the criterion.)

• **Condorcet-consistent rules:** A voting rule r is *Condorcet-consistent* if it always selects the Condorcet winner, whenever one exists. (A Condorcet winner is an alternative that wins in every pairwise election—that is, for any other alternative, more voters prefer the Condorcet winner to this alternative than vice versa.) *Copeland*, *maximin*, *ranked pairs*, *Kemeny*, *Slater*, *Dodgson*, and *voting trees* are all Condorcet-consistent.

• **Majority-consistent rules:** A voting rule r is *majority-consistent* if it always selects the majority winner, whenever one exists. (A majority winner is an alternative that is ranked first by more than half the votes.) Any Condorcet-consistent rule is also majority-consistent, because a majority winner is always a Condorcet winner. In addition, *plurality*, *plurality with runoff*, *STV*, and *Bucklin* are all majority-consistent.

Stackelberg voting game. We now consider the strategic Stackelberg voting game. We use a complete-information assumption: all the voters' preferences are common knowledge. Given this assumption, for any voting rule r, the process where voters vote in sequence can be modeled as an extensive-form game of perfect information, as follows. The game has n stages. In stage j ($j \leq n$), voter j chooses an action from $L(\mathcal{X})$. Each leaf of the tree is associated with an outcome, which is the winner for the profile consisting of the votes that were cast to reach this leaf.

Because the voters' preferences are linear orders (which implies that there are no ties), we can solve the game by backward induction, which results in a unique outcome. We note that this requires only ordinal preferences, that is, we do not need to define utilities. The backward-induction process works as follows. First, for any subprofile of votes by the voters 1 through $n-1$ (that is, any node that is the parent of leaves), there will be a nonempty subset of alternatives that n can make win by casting some vote. She will pick her most preferred one. Now, because we can predict what voter n will do, we take voter $(n-1)$'s perspective: for any subprofile of votes by the voters 1 through $n-2$, there will be a nonempty subset of alternatives that voter $n-1$ can make win by casting some vote (taking into account how voter n will act). She will pick her most preferred one; etc. We continue this process all the way to the root of the tree; the outcome there is called the *backward-induction outcome*.

As noted above, only the ordinal preferences of the voters matter; that is, a voter's preferences correspond to a member of $L(\mathcal{X})$. While votes and preferences both lie in the same set $L(\mathcal{X})$, we must be careful to distinguish between them, because in this context, a voter will sometimes cast a

to specify prior distributions over preferences. Also, naturally, our negative results still apply to more general models, including models allowing for incomplete information.

vote that is different from her true preferences. Nevertheless, we can use $P \in L(\mathcal{X})^n$ to denote a profile of preferences, as well as a profile of votes. For a given voting rule r, let $r(P)$ be the outcome if the *votes* are P; let $SG_r(P)$ be the backward-induction outcome if the *true preferences* are P.[2]

Computing the Backward-Induction Outcome

Even if the outcome of the rule r is easy to compute, it does not follow that the outcome of SG_r is easy to compute. The straightforward backward-induction process described above is very inefficient, because the game tree has $(m!)^n$ leaves.

In this section, we first propose a general dynamic-programming algorithm to compute $SG_r(P)$, for any anonymous voting rule r. Then, we show how to use *compilation functions* (Chevaleyre et al. 2009) to further reduce the time/space-complexity of the dynamic-programming algorithm. These techniques are crucial for obtaining our later experimental results.

The dynamic-programming algorithm still solves the game tree in a bottom-up fashion, but does not need to consider all the different profiles separately. Because r is anonymous, at any stage j of the game, the state (the profile of votes 1 through $j-1$) can be summarized by a vector composed of $m!$ natural numbers, one for each linear order: each number in the vector represents the number of times that the corresponding linear order appears in the $(j-1)$-profile. Formally, for any $j \leq n$, we let the set of these vectors (states) be $S_j = \{(s_1, \ldots, s_{m!}) \in \mathbb{N}_{\geq 0}{}^{m!} : \sum_{i=1}^{m!} s_i = j-1\}$. For any anonymous voting rule r and any $\vec{s} \in S_{n+1}$, let $r(\vec{s})$ be the winner for any profile that corresponds to $\vec{s}$ (because r is anonymous, the winner only depends on the vector $\vec{s}$). More generally, for arbitrary S_j, the algorithm computes a labeling function g that maps each state $\vec{s} \in S_j$ to the alternative representing the backward-induction outcome of the subgame whose root corresponds to $\vec{s}$.

Algorithm 1
Input. $P = (V_1, \ldots, V_n)$ and an anonymous voting rule r.
Output. $SG_r(P)$.
1. For j from $n+1$ to 1, do Step 2.
2. For any state $\vec{s} \in S_j$, do
  2.1 If $j = n+1$, then let $g(\vec{s}) = r(\vec{s})$.
  2.2 If $j < n+1$, then let $\vec{e}^* \in \arg\min_{\vec{e} \in E} \text{rank}(V_j, g(\vec{s}+\vec{e}))$, where E consists of all vectors that are composed of $m!-1$ zeroes and only one 1, and $\text{rank}(V_j, g(\vec{s}+\vec{e}))$ is the position of $g(\vec{s}+\vec{e})$ in V_j. (Thus, e^* corresponds to an optimal vote for j.) Then, let $g(\vec{s}) = g(\vec{s}+\vec{e}^*)$.
3. Output $g((0, \ldots, 0))$.

Analysis. For any $j \leq n$, $|S_j| = \binom{j+m!-2}{m!-1}$ (this is a basic combinatorial result, see e.g. (Bender and Williamson 2006)). To analyze the runtime of the algorithm, we note that the total number of states considered is $\sum_{j=1}^{n+1} \binom{j+m!-2}{m!-1}$, which is $O((n+1)^{m!+1})$; in each state, we need to consider $m!$ vectors $\vec{e}$, resulting in a total bound of $O(m!(n+1)^{m!+1})$. To analyze the space requirements of the algorithm, we note that we only need to keep the last stage $j+1$ and the current stage j in memory, so that the maximum number of states in memory is $\binom{n+m!-1}{m!-1} + \binom{n+m!-2}{m!-1}$, which is $O((n+1)^{m!})$. Therefore, when m is bounded above by a constant, Algorithm 1 runs in polynomial time (using polynomial space).

However, when there is no upper bound on m, Algorithm 1 runs in exponential time and uses exponential space. We conjecture that for many common voting rules (e.g., plurality), computing SG_r is PSPACE-hard, but we have not managed to obtain any such result yet.[3]

Compilation. In the step corresponding to stage j in Algorithm 1, a very large set S_j is used to keep track of all possible $m!$-dimensional vectors whose entries sum to exactly $j-1$, representing the possible states. While it may be necessary to have this many states for anonymous rules in general, it turns out that for specific rules like plurality or veto, we need far fewer states to represent the profiles, because many of the states in Algorithm 1 will be equivalent *for the specific rule*. For example, if we have so far received only a single vote $a \succ b \succ c$, this in general is not equivalent to having received only a single vote $a \succ c \succ b$. However, if the rule is plurality, these states are equivalent.

Pursuing this idea, for any anonymous voting rule r, we can ask the following questions. (1) *What is the smallest set of states needed for stage j?* (2) *How can we incorporate smaller sets of states into Algorithm 1?*

The answer to question (1) corresponds to the *compilation complexity* of r, a concept introduced by Chevaleyre et al. (2009). For any $k, u \in \mathbb{N}$ with $k+u = n$, the compilation complexity $C_{m,k,u}(r)$ is defined to be the smallest number of bits needed to represent all "effectively different" k-profiles, when there are u remaining votes and the winner is chosen by using r. (Two k-profiles are "effectively the same" if, for any profile of u votes that we may add to them, they result in the same outcome.) It follows that, if we tailor Algorithm 1 to a specific rule r, the size of the smallest possible set of states for stage j is between $2^{C_{m,j-1,n-j+1}(r)-1}$ and $2^{C_{m,j-1,n-j+1}(r)}$. Chevaleyre et al. (2009) also studied the compilation complexity for some common voting rules.

Now we turn to address question (2). Suppose that we have already determined that we can use a smaller set of states. In order to modify the dynamic-programming algorithm to use this smaller set of states, for step (2.2) we must have a function that takes a state in S_j and a vote V as inputs, and outputs a state in S_{j+1}; moreover, this function must be easy to compute. Fortunately, the *compilation functions* designed for some common voting rules in (Chevaleyre et al. 2009; Xia and Conitzer 2010), which map each profile to a string (state), can serve as such functions. For example, the compilation function for plurality simply counts how often each alternative has been ranked first, and this is easy to up-

[2] Of course, because it is a function from profiles of linear orders to alternatives, SG_r can also be interpreted as a voting rule, though there is a significant risk of confusion in doing so. We note that even if r is anonymous, SG_r (as a voting rule) is not necessarily anonymous (the order of the voters matters).

[3] We have obtained a PSPACE-hardness result for a not-so-common rule with a different type of voter preferences, which thus falls somewhat outside of the setting described so far. We omit it due to the space constraint.

date. More generally, we can modify Algorithm 1 for any specific rule r as follows. Let $f^r_{m,k,u}$ be a compilation function for r. For any $j \leq n$, we let $S_j = f^r_{m,k,u}(L(\mathcal{X})^{j-1})$, that is, the set of all "compressed" $(j-1)$-profiles. Then, in step (2.2), for each given state $\vec{s} \in S_j$ and each[4] given vote $V \in L(\mathcal{X})$, the next state (which lies in S_{j+1}) is computed by applying the compilation function $f^r_{m,k,u}$ to the combination of $\vec{s}$ and V. Among these resulting states, we again find voter j's most-preferred outcome.

Illustration. Let us illustrate how the use of compilation functions helps reduce the time and space requirements of Algorithm 1 for the nomination rule. In this case, for any $j \leq n$, let $S_j = \mathcal{X}$, and let f^{Nom} be the following compilation function. For any profile P, let $f^{\text{Nom}}(P)$ be the first alternative (according to the order $c_1 > \ldots > c_m$) that has been nominated (is ranked first in some vote in P). For any profile P and any vote V, $f^{\text{Nom}}(P \cup \{V\})$ can be easily computed from $f^{\text{Nom}}(P)$ and V, by determining which of $f^{\text{Nom}}(P)$ and the alternative ranked in the top position in V is earlier in the order. (As in the case of plurality, we do not need to consider every vote V: we only need to consider which alternative is ranked first.) Because $|S_j| = m$ for all j in this case, it follows that Algorithm 1 (using f^{Nom}) runs in polynomial time for the nomination rule.

Proposition 1 *SG_{Nom} can be computed in polynomial time (and space) by Algorithm 1 (using f^{Nom}).*

For other, more common voting rules, the runtime of the dynamic-programming algorithm is also significantly reduced by using compilation functions, though it remains exponential. For example, for plurality and veto, the time/space complexity of our approach is $O(n^m)$, which allows us to conduct the simulation experiments (later in the paper) much more efficiently.

Paradoxes

In this section, we investigate whether the strategic behavior described above will lead to undesirable outcomes. It turns out that it can. Our main theorem is a general result that applies to many anonymous voting rules. We will show that, for such a rule, there exists a profile that has two types of paradox associated with it in the backward-induction outcome: first, the winner loses all but one of its pairwise elections; second, the winner is ranked somewhere in the bottom two positions in almost every voter's true preferences. For the second type of paradox, we will show that the number of exceptions (voters who rank the winner higher) is closely related to a parameter called the *domination index*. The domination index of a voting rule r that satisfies *non-imposition* (that is, any alternative is the winner under *some* profile) is the smallest number q such that any coalition of $\lfloor n/2 \rfloor + q$ voters can make any given alternative win (no matter how the remaining voters vote) under r. We note that the domination index is always well defined for any rule that satisfies non-imposition, and is at least 1.

[4]For some rules, we do not need to consider every vote: for example, under plurality, we do not need to consider both $a \succ b \succ c$ and $a \succ c \succ b$.

Definition 1 *For any voting rule r that satisfies non-imposition, and any $n \in \mathbb{N}$, we let the* domination index *$DI_r(n)$ be the smallest number q such that for any alternative c, and for any subset of $\lfloor n/2 \rfloor + q$ voters, there exists a profile P for these voters, such that for any profile P' for the remaining voters, $r(P, P') = c$.*

The domination index DI_r is closely related to the *anonymous veto function* $\text{VF}_r : \{1, \ldots, n\} \to \{0, \ldots, m\}$ (Definition 10.4 in (Moulin 1991)), defined as follows. $\text{VF}_r(i)$ is the largest number $j \leq m-1$ such that any coalition of i voters can veto any subset (that is, make sure that none of the alternatives in the set is the winner) of no more than j alternatives. We note that the domination index $\text{DI}_r(n)$ for a voting rule r is the smallest number q such that $\text{VF}_r(\lfloor n/2 \rfloor + q) = m-1$ (that is, any coalition of size $\lfloor n/2 \rfloor + q$ can veto any set of $m-1$ alternatives).

The next proposition gives bounds on the domination index for some common voting rules.

Proposition 2 *$DI_{Nom} = \lceil n/2 \rceil$. For any positional scoring rule r, $DI_r \leq \lfloor n/2 \rfloor - \lfloor n/m \rfloor$. For any majority-consistent voting rule r (*e.g.*, any Condorcet-consistent rule, plurality, plurality with runoff, Bucklin, or STV), $DI_r(n) = 1$.*

The next lemma provides a sufficient condition for an alternative not to be the backward-induction winner. It says that if there is a coalition of size $k \geq \lfloor n/2 \rfloor + \text{DI}_r(n)$ who all prefer c to d, and another condition holds, then d cannot win.[5] For any alternative $c \in \mathcal{X}$ and any $V \in L(\mathcal{X})$, we let $\text{Up}(c, V)$ denote the set of all alternatives that are ranked higher than c in V.

Lemma 1 *Let P be a profile. An alternative d is not the winner $SG_r(P)$ if there exists another alternative c and a subprofile $P_k = (V_{i_1}, \ldots, V_{i_k})$ of P that satisfies the following conditions: 1. $k \geq \lfloor n/2 \rfloor + DI_r(n)$, 2. $c \succ d$ in each vote in P_k, 3. for any $1 \leq j_1 < j_2 \leq k$, $Up(c, V_{i_{j_1}}) \supseteq Up(c, V_{i_{j_2}})$.*

Proof. Let $D_k = \{i_1, \ldots, i_k\}$. Since $k \geq \lfloor n/2 \rfloor + \text{DI}_r(n)$, this coalition of voters can guarantee that any given alternative be the winner under r, if they work together. Let $P^*_k = (V^*_{i_1}, \ldots, V^*_{i_k})$ be a profile that can guarantee that c be the winner under r. That is, for any profile P' for the other voters ($\{1, \ldots, n\} \setminus D_k$), we have $r(P^*_k, P') = c$. For any $j \leq k$, we let $D'_{i_j} = \{1, \ldots, i_j\} \setminus D_k$—that is, the first i_j voters, except those in the coalition D_k. For any $j \leq k$, we let $P^*_j = (V^*_{i_1}, \ldots, V^*_{i_j})$. That is, P^*_j consists of the first j votes in P^*_k. For any $i \leq n-1$ and any pair of profiles P_1 (consisting of i votes) and P_2 (consisting of $n-i$ votes), we let $SG_r(P_2 : P_1)$ denote the backward-induction winner of the subgame of the Stackelberg voting game in which voters 1 through i have already cast their votes P_1, and the true

[5]This may seem trivial because the coalition can guarantee that c wins if they work together. However, we have to keep in mind that the members of the coalition each pursue their own interest. For example, it may be the case that whenever the second-to-last voter in the coalition votes in a way that enables the last voter in the coalition to make c the winner, it also enables this last voter to make e the winner, which this last voter prefers—but the second-to-last voter actually prefers d to e, and therefore votes to make d win instead. We need the extra condition to rule out such examples.

preferences of voters $i+1$ through n are as in P_2. We prove the following claim by induction.

Claim 1 *For any $j \leq k$ and any profile P'_{i_j} for the voters in D'_{i_j}, $SG_r((V_{i_j}, V_{i_j+1}, \ldots, V_n) : P'_{i_j}, P^*_{j-1}) \succeq_{V_{i_j}} c$.*

Claim 1 states that for any $j \leq k$, if voters $i_1, \ldots, i_{j-1}$ have already voted as in P^*_j, and voter i_j will vote next, then the backward-induction outcome of the corresponding subgame must be (weakly) preferred to c by voter i_j.

Proof of Claim 1: The proof is by (reverse) induction on j. First, we consider the base case where $j = k$. If voter i_k casts $V^*_{i_k}$, then the winner is c, because the subprofile P^*_k will guarantee that c wins. Voter i_k will only vote differently if it results in at least as good an outcome for her as c. Therefore, the claim holds for $j = k$.

Now, suppose that for some j', the claim holds for $j' \leq j \leq k$. We will now show that it also holds for $j = j' - 1$. Let c' be the backward-induction outcome when voter $i_{j'-1}$ submits $V^*_{i_{j'-1}}$. By the induction hypothesis, we have that $c' \succeq_{V_{i_{j'}}} c$. That is, voter $i_{j'}$ (weakly) prefers c' to c. We recall that $\text{Up}(c, V_{i_{j'-1}}) \supseteq \text{Up}(c, V_{i_{j'}})$, which means that c' is also (weakly) preferred to c by voter $i_{j'-1}$. This means that voter $i_{j'-1}$ can guarantee that the outcome be at least as good as c for her. She will only vote differently from $V^*_{i_{j'-1}}$ if it results in at least as good an outcome for her as c' (which is at least as good as c already). Therefore, the claim also holds for $j' - 1$, and Claim 1 follows by induction. □

Letting $j = 1$ in Claim 1, we have that $SG_r(P) \succeq_{V_{i_1}} c$. Therefore, $d \neq SG_r(P)$ (because $c \succ_{V_{i_1}} d$). This completes the proof of Lemma 1. □

We are now ready to present our main theorem. We note that this theorem does not depend on the tie-breaking mechanism used in the rule.

Theorem 1 *For any voting rule r that satisfies non-imposition, and any $n \in \mathbb{N}$, there exists a profile P such that $SG_r(P)$ is ranked somewhere in the bottom two positions in $n - 2\text{DI}_r(n)$ of the votes, and, if $\text{DI}_r(n) < n/4$, then $SG_r(P)$ loses to all but one alternative in pairwise elections.*

Proof. The proof is constructive. Let $P = (V_1, \ldots, V_n)$ be the profile (the voters' true preferences) defined as follows.

$$V_1 = \ldots = V_{\lfloor n/2 \rfloor - \text{DI}_r(n)} = [c_3 \succ \ldots \succ c_m \succ c_1 \succ c_2]$$
$$V_{\lfloor n/2 \rfloor - \text{DI}_r(n)+1} = \ldots = V_{\lfloor n/2 \rfloor + \text{DI}_r(n)} = [c_1 \succ c_2 \succ c_3 \succ \ldots \succ c_m]$$
$$V_{\lfloor n/2 \rfloor + \text{DI}_r(n)+1} = \ldots = V_n = [c_2 \succ c_3 \succ \ldots \succ c_m \succ c_1]$$

We now use Lemma 1 to prove that $SG_r(P) = c_1$. First, we let $k = \lfloor n/2 \rfloor + \text{DI}_r(n)$, and let P_k be the first k votes. It follows from Lemma 1 (letting $c = c_1$ and $d = c_2$) that $c_2 \neq SG_r(P)$. Next, for any $c' \in \mathcal{X} \setminus \{c_1, c_2\}$, we let $k = \lceil n/2 \rceil + \text{DI}_r(n)$ and let P_k be the last k votes, that is, $P_k = (V_{\lfloor n/2 \rfloor - \text{DI}_r(n)+1}, \ldots, V_n)$. By Lemma 1 (letting $c = c_2$ and $d = c'$), we have that $c' \neq SG_r(P)$. It follows that $SG_r(P) = c$.

In P, c_1 is ranked somewhere in the bottom two positions in $n - 2\text{DI}_r(n)$ votes (the first $\lfloor n/2 \rfloor - \text{DI}_r(n)$ votes and the last $\lceil n/2 \rceil - \text{DI}_r(n)$ votes). If $\text{DI}_r(n) < n/4$, then $2\text{DI}_r(n) < n/2$, which means that c_1 will lose to any other alternative (except c_2) in pairwise elections. □

Combining Proposition 2 and Theorem 1, we obtain the following corollary for common voting rules.

Corollary 1 *Let r be any majority-consistent rule and let $n \geq 5$. There exists a profile P such that $SG_r(P)$ is ranked somewhere in the bottom two positions in $n - 2$ votes; moreover, $SG_r(P)$ loses to all but one alternative in pairwise elections. (This holds regardless of how ties are broken.)*

While this is a strong paradox already, it is sometimes possible to obtain even stronger paradoxes if we restrict attention to individual rules. We illustrate this on the plurality and nomination rules. We recall that ties are broken in the order $c_1 > \ldots > c_m$. We only give some proof sketches showing the paradoxical profile for Proposition 3. The proofs are omitted due to the space constraint.

Proposition 3 *For any $m \geq 3$, if n is even, then there exists an n-profile P such that $SG_{Plu}(P)$ is ranked somewhere in the bottom two positions in every voter's true preferences; moreover, all voters prefer all but one other alternatives to $SG_{Plu}(P)$ (that is, $SG_{Plu}(P)$ is* Pareto-dominated *by all but one other alternatives). (This assumes ties are broken in the order $c_1 > \ldots > c_m$.)*

Proof sketch. We let P be an n-profile consisting of the following votes.

$$V_1 = \ldots = V_{n/2} = [c_3 \succ c_4 \succ \ldots \succ c_m \succ c_1 \succ c_2]$$
$$V_{n/2+1} = \ldots = V_n = [c_2 \succ c_3 \succ \ldots \succ c_m \succ c_1]$$

It can be shown that $SG_{Plu}(P) = c_1$. □

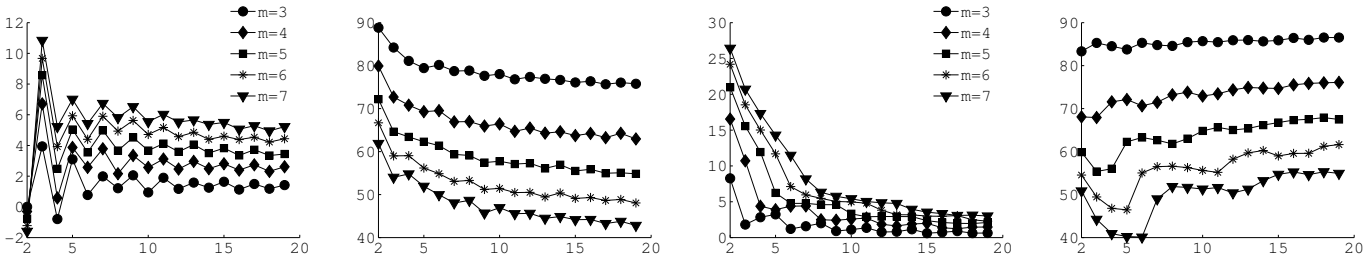
Proposition 4 *For any $m \in \mathbb{N}$, if n is odd, then there exists an n-profile P such that $SG_{Plu}(P)$ is ranked somewhere in the bottom $\lceil 2m/(n+1) \rceil + 2$ positions in every vote's true preferences. (This assumes ties are broken in the order $c_1 > \ldots > c_m$.)*

We recall that the domination index for Nom is $\lceil n/2 \rceil$. Therefore, Theorem 1 does not imply any real paradox for Nom. However, paradoxes for Nom can still be obtained directly, as the following proposition shows.

Proposition 5 *For any m, n, there exists a profile P such that in each vote, $SG_{Nom}(P)$ is ranked somewhere in the bottom $(\lceil m/n \rceil + 1)$ positions in each voter's true preferences; moreover, $SG_{Nom}(P)$ loses in all pairwise elections (that is, it is a* Condorcet-loser*).*

Experimental results

In the previous section, we showed that the backward-induction solution to the Stackelberg voting game is socially undesirable for some profiles. We may ask ourselves whether such profiles are common, or just isolated instances that are not very likely to happen in practice. To answer this question, we will compare the backward-induction winner $SG_r(P)$ to a benchmark outcome—namely, the alternative $r(P)$ that would win under r if all voters vote truthfully. This may seem like a difficult benchmark to achieve, because often strategic behavior comes at a cost (cf. price of anarchy, first-best vs. second-best in mechanism design, etc.). Nevertheless, in the experiments that we describe in this section, it turns out that in randomly chosen profiles, in fact,



(a) (b) (c) (d)

Figure 1: The x-axis gives the number of voters (n); the y-axis gives the percentage of voters. In each case we consider various numbers of alternatives (m). (a) The percentage of voters who prefer the SG_r winner to the r winner minus the other way around, under plurality. (b) The percentage of profiles for which the SG_r winner and the r winner are the same, under plurality. (c) The percentage of voters who prefer the SG_r winner to the r winner minus the other way around, under veto. (d) The percentage of profiles for which the SG_r winner and the truthful r winner are the same, under veto. Please note the different scales on the y-axis for (a) and (c).

slightly more voters prefer the backward-induction outcome $SG_r(P)$ to the truthful outcome $r(P)$ than vice versa!

The setup of our experiment is as follows. We study the plurality and veto rules (these are the easiest to scale to large numbers of voters, because they have low compilation complexity).[6]

For any m, n, and $r \in \{\text{Plurality}, \text{Veto}\}$, our experiment has 25,000 iterations. In each iteration, we perform the following three steps. 1. In iteration j, an n-profile P_j is chosen uniformly at random from $L(\mathcal{X})^n$. 2. We calculate $SG_r(P_j)$ using Algorithm 1 (with a compilation function to reduce time/space-complexity), and we calculate $r(P_j)$. 3. We then count the number of voters in this profile P that prefer $SG_r(P)$ to $r(P)$ (according to their true preferences in P), denoted by n_1, and vice versa, denoted by n_2. If $SG_r(P) = r(P)$, then $n_1 = n_2 = 0$.

For each m, n, r, we calculate the total percentage (across all 25,000 iterations) of voters that prefer the backward-induction winner for their profile to the winner under truthful voting for their profile, that is, $p_1 = \sum_{j=1}^{25000} n_1^j/(25000n)$. We also compute $p_2 = \sum_{j=1}^{25000} n_2^j/(25000n)$. We note that it is not necessarily the case that $p_1 + p_2 = 1$, because if $SG_r(P) = r(P)$, then $n_1 = n_2 = 0$. Let $p_3 = 1-p_1-p_2$ be the percentage of profiles for which the backward-induction (SG_r) winner coincides with the truthful (r) winner. We are primarily interested in $p_1 - p_2$.

The results are summarized in Figure 1. First, from (a) and (c) it can be observed that for plurality and veto, perhaps surprisingly, on average, more voters prefer the backward-induction winner to the winner under truthful voting than vice versa. Generally, the difference becomes smaller when n increases; the difference is larger when m is larger; and the percentage seems to converge to some limit as $n \to \infty$. Second, from (b) and (d) it can be observed that the percentage of profiles for which the two winners coincide is smaller for larger values of m; the percentage is decreasing in the number of voters n for plurality, but increasing for veto.

[6]We also investigated other rules. It appears that they may lead to similar results, though it is difficult to say this with high confidence because we can only solve for the backward-induction outcome for small numbers of voters.

Future Work

There are several directions for future research. First, is it possible to design algorithms that compute the backward-induction outcome efficiently, even for rules with high compilation complexity and with many alternatives? We conjecture that without any bound on the number of alternatives, PSPACE-hardness results lie in waiting. If so, what implications does this have for practical strategic voting in the Stackelberg voting game? Second, is it possible to more generally characterize the circumstances under which the backward-induction outcome is "better" than the truthful-voting outcome? If so, can this lead to practical recommendations about when Stackelberg voting should be encouraged?

References

Battaglini, M. 2005. Sequential voting with abstention. *Games and Economic Behavior* 51:445–463.

Bender, E. A., and Williamson, S. G. 2006. *The Foundations of Combinatorics with Applications*. Dover.

Chevaleyre, Y.; Lang, J.; Maudet, N.; and Ravilly-Abadie, G. 2009. Compiling the votes of a subelectorate. In *IJCAI*, 97–102.

Dekel, E., and Piccione, M. 2000. Sequential voting procedures in symmetric binary elections. *JPE* 108:34–55.

Desmedt, Y., and Elkind, E. 2010. Equilibria of plurality voting with abstentions. In *EC*.

Farquharson, R. 1969. *Theory of Voting*. Yale University Press.

Gibbard, A. 1973. Manipulation of voting schemes: a general result. *Econometrica* 41:587–602.

McKelvey, R. D., and Niemi, R. G. 1978. A multistage game representation of sophisticated voting for binary procedures. *JET* 18(1):1–22.

Moulin, H. 1979. Dominance solvable voting schemes. *Econometrica* 47:1337–51.

Moulin, H. 1991. *Axioms of Cooperative Decision Making*. Cambridge University Press.

Satterthwaite, M. 1975. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *JET* 10:187–217.

Sloth, B. 1993. The theory of voting and equilibria in noncooperative games. *Games and Econ. Behavior* 5:152–169.

Xia, L., and Conitzer, V. 2010. Compilation complexity of common voting rules. In *AAAI*.