Information and Computation 209 (2011) 89–107

Contents lists available at ScienceDirect

Information and Computation

journal homepage: www.elsevier.com/locate/ic

The shield that never was: Societies with single-peaked preferences are more open to manipulation and control ☆

Piotr Faliszewski [a,*], Edith Hemaspaandra [b], Lane A. Hemaspaandra [c], Jörg Rothe [d]

[a] Department of Computer Science, AGH University of Science and Technology, 30-059 Kraków, Poland
[b] Department of Computer Science, Rochester Institute of Technology, Rochester, NY 14623, USA
[c] Department of Computer Science, University of Rochester, Rochester, NY 14627, USA
[d] Institut für Informatik, Heinrich-Heine-Universität Düsseldorf, 40225 Düsseldorf, Germany

ARTICLE INFO

Article history:
Received 17 September 2009
Revised 7 July 2010
Available online: 14 October 2011

Keywords:
Algorithms
Complexity theory
Computational social choice
Control and manipulation of elections
Multiagent systems
Preference aggregation

ABSTRACT

Much work has been devoted, during the past 20 years, to using complexity to protect elections from manipulation and control. Many "complexity shield" results have been obtained—results showing that the attacker's task can be made NP-hard. Recently there has been much focus on whether such worst-case hardness protections can be bypassed by frequently correct heuristics or by approximations. This paper takes a very different approach: We argue that when electorates follow the canonical political science model of societal preferences the complexity shield never existed in the first place. In particular, we show that for electorates having single-peaked preferences, many existing NP-hardness results on manipulation and control evaporate.

1. Introduction

Elections represent a crucial process in human societies. They are also used in multiagent systems. For example, elections have been proposed as a mechanism for collaborative decision-making in such multiagent system contexts as recommender systems/collaborative filtering [39] and planning [19,20]. The importance of elections explains why elections are studied intensely over a wide range of fields, including political science, mathematics, social choice, artificial intelligence, economics, and operations research. Formally, an election system takes as input a set of candidates or alternatives and a set of votes, and outputs a subset of the candidates as the winner(s). The votes are over some domain, typically the linear orders over the candidates or the 0-1 "approval vectors" over the candidates.

"Control" and "manipulation" are terms-of-art for two types of manipulative actions on elections. We provide definitions in Section 2. However, briefly put, control refers to attempts to make a given candidate win [6] or not win [33] an election by actions such as adding or deleting voters or candidates. And manipulation refers to attempts to make a given candidate win [4,5] or not win [16] by some coalition of voters who strategically change their votes. There is a large literature, started

☆ Supported in part by DFG grants RO-1202/{11-1, 12-1}, NSF grants CCF-0426761, IIS-0713061, and CCF-0915792, the European Science Foundation's EUROCORES program LogICCC, Polish Ministry of Science and Higher Education grant N-N206-378637, AGH University of Science and Technology grant 11.11.120.865, the Homing/Powroty program of the Foundation for Polish Science, and Friedrich Wilhelm Bessel Research Awards to Edith Hemaspaandra and Lane A. Hemaspaandra from the Alexander von Humboldt Foundation. Work done in part during visits by the first three authors to Heinrich-Heine-Universität Düsseldorf and by the first and fourth authors to the University of Rochester. A preliminary version of this paper appeared in the Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge, July 2009.

* Corresponding author.
E-mail address: faliszew@agh.edu.pl (P. Faliszewski).
URLs: http://home.agh.edu.pl/~faliszew (P. Faliszewski), http://www.cs.rit.edu/~eh (E. Hemaspaandra), http://www.cs.rochester.edu/u/lane (L.A. Hemaspaandra), http://ccc.cs.uni-duesseldorf.de/~rothe (J. Rothe).

0890-5401/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2010.09.001

by the insightful contributions of Bartholdi, Orlin, Tovey, and Trick [4–6], on choosing election systems that make control and manipulation NP-hard, i.e., on choosing election systems that seek to make control and manipulation computationally prohibitive (see the survey [25]). Recently, there has been a flurry of work seeking to bypass worst-case manipulation hardness results by frequently correct heuristics or approximation algorithms (as some pointers into that literature, see, e.g., [13,15,28,41,44,45]).

The present paper takes a radically different approach. We study elections where the vote set must be "single-peaked." We will discuss single-peakedness in more detail after stating our main contributions. But, simply put, in the model in which votes are linear orders single-peakedness means that there is some linear ordering of the candidates relative to which each voter's preferences always increase, always decrease, or first increase and then decrease. In the model in which votes are approval vectors, single-peakedness means there is some linear ordering of the candidates relative to which each voter's approved candidates are contiguous. Single-peaked preferences, introduced by Black [9,10], model societies that are heavily focused on one issue (taxes, war, etc.). The single-peaked framework is so central to political science that it has been described as "*the* canonical setting for models of political institutions" [30]. Indeed, it is typically the model of societal voting first covered in an introductory course on positive (i.e., theoretical) political science. This paper's main contributions are the following:

1. We introduce the study of single-peakedness for approval-voting elections. Approval-voting elections are elections where voters vote by approval vectors, and the candidate(s) with the highest number of approvals win.
2. In Section 3 we show that for both approval voting and plurality voting, many election control problems known to be NP-hard in the general case have polynomial-time algorithms in the single-peaked case.
3. In Section 4 we show that many election manipulation problems known to be NP-hard in the general case have polynomial-time algorithms in the single-peaked case.
   However, in Section 4 we also show that many manipulation problems remain NP-hard even when restricted to the single-peaked case. In this result we were inspired by the path-setting work of Walsh. Walsh [43] showed that single transferable vote, for at least three candidates and weighted votes, remains NP-hard to manipulate even in the single-peaked case.
4. In Theorem 4.2 we show that in the single-peaked case in 3-veto elections (i.e., each voter votes against three candidates and for all others) manipulation is in P for up to four candidates, is NP-hard for five candidates, and is in P for six or more candidates. This shows that, contrary to intuition, even for natural systems there are cases where increasing the number of candidates decreases the complexity.

We mostly defer discussion of related work until after our results, as the related work will then have more context and definitions to draw on. However, we mention now an issue that may at this point worry readers familiar with "median voting." "Median voting" and "generalized median voter schemes" are known not to give voters incentives to vote insincerely, for single-peaked electorates (see [3,38] and the references therein for definitions and discussion). One might naturally wonder whether single-peaked electorates should use only median voting. However, in real-life scenarios voting rules are typically fixed (e.g., to be plurality or approval) and cannot be changed when one suspects the electorate to be single-peaked. In addition, lack of incentive to vote insincerely regarding manipulation does not say anything about a rule's resistance to control.

2. Preliminaries

2.1. Elections and preferences

An *election* consists of a set C of candidates and a collection V of votes. We will consider two different models for votes. One is that each vote is a vector (an *approval vector*) from $\{0, 1\}^{\|C\|}$, denoting approval (1) or disapproval (0) for each candidate. The other model is that each vote is a linear order over the candidates, e.g., Bob $>$ Alice $>$ David $>$ Carol. By linear order we mean a strict, linear order—a complete, transitive, antisymmetric relation. An *election system* is a mapping that takes as input a candidate set C and a set V of votes over that candidate set, and outputs an element of 2^C, i.e., outputs which candidates are *winners* of the election. We, like Bartholdi et al. [6], do not expressly forbid elections with no winners. However, all the natural election systems discussed in this paper have the property that there is always at least one winner when there is at least one candidate. Except where we explicitly state otherwise, V is a list of votes (ballots), so if three votes are the same, they will appear three times in the list. We will use *succinct input* [23] to describe the quite different input model in which each preference that is held by one or more voters appears just once in the list and is accompanied by a binary number stating how many voters have that preference.

The election systems of most interest to us in this paper are the following ones. In *approval voting*, voters vote by approval vectors, and whichever candidate(s) get the most approvals are the winner(s). A *scoring protocol* election, which is always defined for a specific number m of candidates, is specified by a *scoring vector*, $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m) \in \mathbb{N}^m$, satisfying $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_m$. Votes are linear orders. Each vote contributes α_1 points to that vote's most preferred candidate, α_2 points to that vote's second most preferred candidate, and so on. And whichever candidate(s) get the most total points are the winner(s). Among the most important scoring protocols for m candidates are plurality, with $\alpha = (1, \overbrace{0, \ldots, 0}^{m-1})$, j-veto

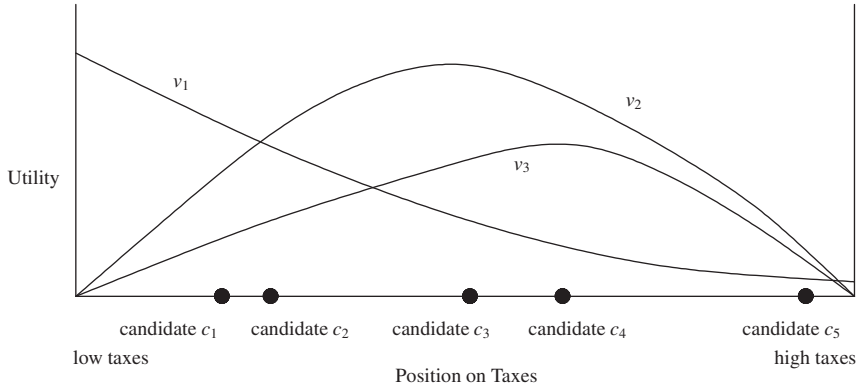

Fig. 1. Single-Peaked Preference Motivation: Utility Curves for Three Voters.

($j \leq m$), with $\alpha = (\overbrace{1, \ldots, 1}^{m-j}, \overbrace{0, \ldots, 0}^{j})$, and Borda, with $\alpha = (m-1, m-2, \ldots, 0)$. For m candidates, j-approval and $(m-j)$-veto are the same system. We will also speak of the veto system for an unbounded number of candidates, by which we mean that on inputs with m candidates, each voter gives 1 point to all candidates other than her least favorite candidate and 0 points to her least favorite candidate. We will similarly also speak of plurality for an unbounded number of candidates, by which we mean that on inputs with m candidates, each voter gives 0 points to all candidates other than her favorite candidate and 1 point to her favorite candidate. And the general case of approval voting is always for an unbounded number of candidates.

Let us fix some notation for the election systems defined above. Let (C, V) be an election and let c be a candidate in C. By $score_{(C,V)}(c)$ we mean the score of c in election (C, V), i.e., the number of points c receives under plurality, veto, or some given scoring protocol and the number of c's approvals under approval voting. It will always be clear from the context which election system the score refers to. If the candidate set C is clear from the context, we simply write $score_V(c)$.

2.2. Single-peaked preferences

A collection V of votes, each vote v_i being a linear order $>_i$ over C, is said to be *single-peaked* exactly if there exists a linear order over C, call it L, such that for each triple of candidates c, d, and e, it holds that:

$$(c\, L\, d\, L\, e \vee e\, L\, d\, L\, c) \Longrightarrow (\forall i)\,[c >_i d \Longrightarrow d >_i e].$$

This is simply a formal way of saying that with respect to L, each voter's degree of preference just rises, just falls, or rises to a peak and then falls. The loose intuition behind this is captured in Fig. 1. If we imagine an electorate completely focused on one issue (say taxes), with each person having a single-peaked—in the natural analogous sense of that notion applied to curves—utility curve, one gets precisely this notion, give or take the issue of ties. Note that different voters can have different utility curves, as they do in this example. In Fig. 1, the preferences of v_1 would be $c_1 > c_2 > c_3 > c_4 > c_5$, of v_2 would be $c_3 > c_4 > c_2 > c_1 > c_5$, and of v_3 would be $c_4 > c_3 > c_2 > c_1 > c_5$.

The seminal study of single-peaked preferences was done by Black [9], and that work and many subsequent studies, e.g., [10,17,35,37,40], argued that single-peaked preferences (in the unidimensional spatial model) are a broadly useful model of electoral preferences that captures many important settings. Of course, issues that are multidimensional—as many issues are—are typically not captured by single-peaked preferences (by which we always mean the unidimensional case). And even in a society that is completely focused on one issue, some maverick individuals may focus on other issues, so one should keep in mind that the single-peaked case is a widely studied but extreme model.

Looking again at Fig. 1, let us imagine that each voter has a utility threshold at which she starts approving of candidates. Then in the model in which votes are approval vectors, we will have that for some linear order L, each voter either disapproves of everyone or approves of precisely a set of candidates that are contiguous with respect to L. Formally, we say a collection V, made up of approval vectors $v_1, v_2, \ldots, v_n$ over the candidate set C, with Approves_i being the set of candidates that v_i approves, is *single-peaked* exactly if there is a linear order over C, call it L, such that for each triple of candidates c, d, and e, it holds that:

$$c\, L\, d\, L\, e \Longrightarrow (\forall i)\,[\{c, e\} \subseteq \text{Approves}_i \Longrightarrow d \in \text{Approves}_i].$$

The reasons that single-peaked approval voting is compellingly natural to study—and the reasons why one should not go overboard and claim that it is a universally appropriate notion—are essentially the same as those, touched on above, involving single-peaked voting with respect to preference orders. In many cases it is natural to assume that there is some unidimensional issue steering society and that each person's range of comfort on that issue is some contiguous segment along that issue's dimension. In such a case, each person's set of approved candidates will form a contiguous block among

the candidates when they are ordered by their positions on that issue. To the best of our knowledge, this is the first paper to study elections based on single-peaked approval vectors.

In our control and manipulation problems, we will for the single-peaked case follow Walsh [43] and take as part of the input a particular linear order of the candidates relative to which the votes are single-peaked. This is arguably natural—as we may view candidates' positions on the issue that defines the entire election as being openly known. However, one may wonder how hard it is, given a set of voters, to tell whether it is single-peaked. For the case of votes being linear orders, Bartholdi and Trick ([7], see also [18,22]; a somewhat related paper is [42]) show by a path-based graph algorithm that the problem is in P. Doignon and Falmagne [18] and Escoffier et al. [22] show how to produce in polynomial time a linear order witnessing the single-peakedness when such an ordering exists. For the case of votes being approval vectors, the literature already has long contained the analogue of both the results just mentioned. In particular, the work of Fulkerson and Gross [29, Sections 5 and 6] and Booth and Lueker [11, Theorem 6] proves, rephrased in our terminology, that in polynomial time—in a certain natural sense, even in linear time—one can determine whether a set of approval vectors is single-peaked, can in such a case efficiently find a linear ordering realizing the single-peakedness, and indeed can implicitly represent all linear orderings realizing the single-peakedness.

Theorem 2.1 ([11,29])**.** *There is a polynomial-time algorithm that, given a collection V of approval vectors over C, produces a linear order L witnessing V's single-peakedness or determines that V is not single-peaked.*

Note that Theorem 2.1 does not seem to follow from the analogous result for linear orders. For example, extending each approval vector to form a linear order with all approved candidates preceding all the disapproved ones and then running the algorithm of Escoffier et al. [22] on the thus transformed input does not work.

2.3. Control and manipulation problems

For an election system $\mathcal{E}$, the *Constructive Control by Adding Candidates* problem is the set of all (C, V, p, k, C'), where V consists of votes over $C \cup C'$, $p \in C$, and $C \cap C' = \emptyset$, such that there is a set $C'' \subseteq C'$ with $\|C''\| \leq k$ for which candidate p is the unique winner under election system $\mathcal{E}$ when the voters in V vote over $C \cup C''$ (i.e., we restrict each voter down to her induced preferences over $C \cup C''$). The *Destructive Control by Adding Candidates* problem is the same except now the question is whether there is such a C'' ensuring that p is *not* a unique winner.

The Constructive/Destructive Control by Deleting Candidates problems are analogous, with inputs of the form (C, V, p, k), where k is a limit on how many candidates from C can be deleted, and it is forbidden to delete p. The Constructive/Destructive Control by Adding/Deleting Voters problems are analogous, with inputs, respectively (C, V, p, k, V') and (C, V, p, k), where V', $V' \cap V = \emptyset$, is a pool of "unregistered" voters, and k denotes the bound on how many voters from V' we can add (Adding case) or how many voters from V we can delete (Deleting case).

Constructive/Destructive Control by Unlimited Adding Candidates is the same as Constructive/Destructive Control by Adding Candidates except there is no "k"; one may add any or all of the members of C'.

Most of these problems were introduced in the seminal control paper of Bartholdi et al. [6], and the remaining ones were introduced in [24,33]. These problems model such real-world situations as introducing a candidate to run to split off another candidate's support; urging an independent candidate to withdraw; spreading rumors that people with outstanding warrants who try to vote will be arrested; and sending vans to retirement homes to drive car-less members of one's party to the voting place.

In each case, the inputs we specified are for the "general" case. For the single-peaked case there will be an additional input, a linear order L over the candidates such that relative to L the election is single-peaked (with respect to all voters and candidates—even those in C' and in V' in problems having those extra sets). If L is not such a linear order, the input is immediately rejected; L's goodness can be easily tested in polynomial time, simply by looking at each vote. As mentioned previously, in assuming that L is given we are following Walsh's [43] natural model—L is the broadly known positioning of the candidates.

Both for linear-order votes ([18], see also [7,22]) and for approval-vector votes (Theorem 2.1) it holds that even if L is not given one can find a good L in polynomial time if one exists. This fact may be comforting to those who would prefer that L not be given. But we caution that a given vote set V may have many valid L's. And for some problems, which valid L one uses may affect the problem and its complexity. Indeed, not having L as part of the input might even open the door to time-sequence issues, e.g., in deletion of candidates/voters, should we ask instead whether single-peakedness should only have to hold after the deletion? (We'd say, ideally, "no.") These issues are reasonable for further control/manipulation-related study (and the issue of whether L is known *a priori* is much studied in the political economy literature already, see, e.g., [1, Section 2.4]). However, we find the Walsh [43] model to be the most natural and compelling. In many cases our proofs work fine if L is not part of the input, and at times we will mention that.

The (constructive coalition weighted) manipulation problem was introduced by Conitzer et al. ([16] and its conference precursors, building on the seminal manipulation papers [4,5]) and takes as input a list of candidates C, a list of nonmanipulative voters each specified by preferences (as a linear order or as an approval vector, depending on our election system) over C and a nonnegative integer weight, a list of the weights of the voters in our manipulating coalition, and the candidate p that our coalition seeks to make a winner. The set of all such inputs for which there is some assignment of preferences to the

manipulators that makes p a winner is the *Constructive Coalition Weighted Manipulation* problem for that election system. The *Constructive Size-k-Coalition Unweighted Manipulation* problem is defined analogously, except with all voters having unit weight and with k being the number of manipulators.

We follow the convention from [4–6] of focusing on the unique-winner case for control and the winner case (i.e., asking whether the candidate can be made *a* winner or be prevented from being *a* winner; following the literature, we will sometimes refer to that as the "nonunique-winner model/case") for manipulation. In many cases we've shown a given result in both models and will sometimes mention that in passing. All of the results of Conitzer, Sandholm, and Lang of relevance to us hold in both models (see Footnote 7 of [16]).

For the single-peaked case, a linear order L is given as part of the input and the manipulating coalition's votes must all be single-peaked with respect to L (as must all other voters) for the input to be accepted.

2.4. Complexity notions for control and manipulation problems

If by a given type of constructive control action we can never change p from not unique winning (not winning) to unique winning (winning) we say the problem is *immune* to that control type. Otherwise we call it *susceptible* to that control type. The destructive case is analogous. If a susceptible problem is in P we call it *vulnerable*. If it is NP-hard we call it *resistant*. A manipulation problem is said to be *vulnerable* if it is in P and *resistant* if it is NP-hard. In each control/manipulation case in this paper where we assert vulnerability (or membership in P), not only is the decision problem in P but also in polynomial time we can produce a successful control/manipulation action if one exists (i.e., the problem is what [33] calls *certifiably vulnerable*). Most of these notions are taken in detail from, or are in the general spirit of, the work of Bartholdi, Orlin, Tovey, and Trick [4–6], as in some cases naturally modified or extended in [33,34].

Almost all NP-hardness proofs of this paper follow via reductions from the well-known NP-complete problem PARTITION (see, e.g., [31]). Specifically, we use the variant of the problem where our input contains a set $\{k_1, \ldots, k_n\}$ of n distinct positive integers that sum to $2K$, and we ask: does there exist a subset $A \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in A} k_i = K$?

3. Control

3.1. Results

We now turn to our results on control of elections. The theme of this paper is that electorates limited to being single-peaked often are simpler to control and manipulate. Intuitively speaking, we will show that the more limited range of vote collections allowed by single-peaked voting (as opposed to general voting) is so restrictive that the reductions showing NP-hardness fall apart. In particular, unless P = NP those reductions centrally need complex collections of votes to work. We will show this by proving that our single-peaked control problems are in P. The techniques we use to prove them in P vary from easy observations to smart greedy schemes to dynamic programming.

The control complexity of approval voting is studied in detail by Hemaspaandra et al. [33] (see [21] regarding limited adding of candidates, see also the survey [8] for an overview of computational properties of approval voting). Among all the adding/deleting control cases defined in this paper (10 in total), precisely two are resistant, and the other eight (all five destructive cases and the three constructive candidate cases) are immune or vulnerable. The two resistant cases are Constructive Control by Adding Voters and Constructive Control by Deleting Voters.

The following theorem shows that both of these complexity shields evaporate for societies with single-peaked preferences. Briefly put, the challenge here is that the set V' of voters to add may be filled with "incomparable" voter pairs—pairs such that regarding their interval with respect to the linear order defining single-peakedness, neither is a subset of the other. So it is not immediately obvious what voters to add. We solve this by a "smart greedy" approach. This involves breaking votes first into broad groups based on where their intervals' right edges fall with respect to just a certain "dangerous" subset of the candidates, and then re-sorting those based on their left edges, and we argue that if any strategy will reach the control goal then this one will.

Theorem 3.1. *For the single-peaked case, approval voting is vulnerable to constructive control by adding voters and to constructive control by deleting voters, in both the unique-winner model and the nonunique-winner model, for both the standard input model and the succinct input model.*

This result holds in our settled model in which the linear order specifying the society's order on the candidates is part of the input, and also holds in the model in which the linear order is not part of the input but rather the question is whether there exists any valid linear order relative to which there is a way of achieving our control goal.

Proof. We will cover the unique-winner cases only. From these it will be completely clear how to handle the nonunique-winner cases. We will cover just the succinct-input cases, as each implies the corresponding standard-input case. Susceptibility is immediately clear by easy examples.

First, let us speak of the societal linear order. In the model in which the (purported) linear order is part of the input, start by doing the obvious polynomial-time check to see whether this linear order truly is valid with respect to the voters V, the

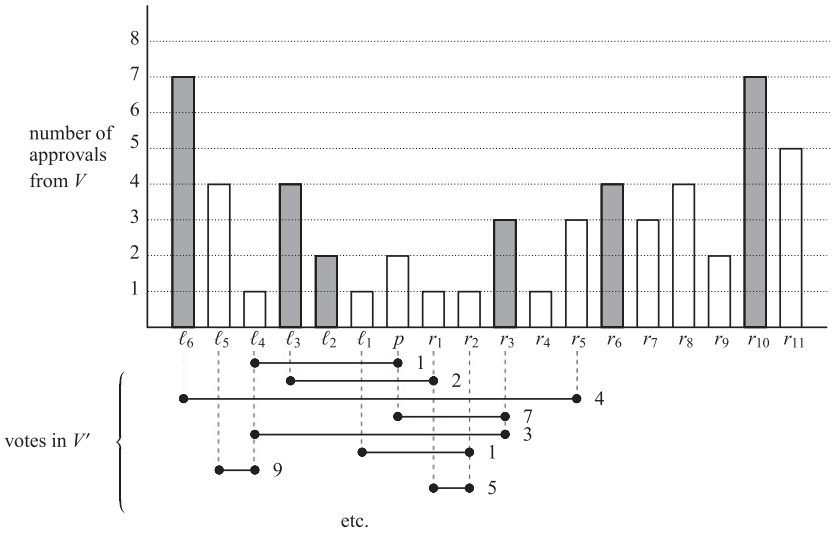

Fig. 2. Bar chart of approvals from V, plus intervals indicating the votes in V' and their multiplicities. The candidates are ordered in the societal linear order and the "dangerous" candidates are marked with shaded bars.

unregistered voters V', and the candidates C. That is, check to ensure that each member of V and V' approves either of no candidates or approves of exactly a collection of candidates that are contiguous with respect to the input linear order. If so, the votes of $V \cup V'$, which are approval vectors over C, are single-peaked with respect to the input societal linear order over C. If not, output "invalid societal linear order" and halt. This very easy observation in effect shows that, for the approval-vector case, checking a given purported societal linear order is in P. That is equally obviously true for the votes-are-linear-orders setting, of course.

If we're in the model in which a linear order must be found, if one exists, relative to which the problem is single-peaked and the given control action is possible, simply use Theorem 2.1 to efficiently find some linear order relative to which our input is single-peaked. If none exists, state that and terminate. If Theorem 2.1 gives us some valid order, it is not hard to see that the algorithm we are going to give in this proof will find a successful control action if and only if there exists any valid societal linear order relative to which a successful control action exists. That is basically because if π_1 and π_2 are valid societal linear orders, the votes are the same either way, and in each case the algorithm we are about to give will succeed exactly if there exists a legal-cardinality collection of unregistered voters that achieves the control goal.

So let us take it that we have the societal linear order, which we will refer to as L. We start with the case of adding voters. This proof will be done rather visually. And although we will implicitly be sketching a general proof, we will keep close to heart an example, as that will make the idea of the proof clearer. Let p be the preferred candidate—the one we seek to make a unique winner. Fig. 2 shows a representation of the initial state. In this figure, for display purposes we have renamed all the candidates in a way that echos the societal linear order L, i.e., the candidates are each labeled with respect to whether they are left or right of p in L, and are ordered following the societal order. The bars represent the total number of approvals each candidate gets over all the votes contained in V. Since we're in the succinct-input case, we of course in computing that take into account the multiplicities, e.g., if candidate r_{10} is approved of in one ordering that occurs with multiplicity 4 and one that occurs with multiplicity 3, and in no others, then the bar for r_{10} would show that it gets 7, not 2, approvals. The intervals below the bar graph reflect the votes in V'. Note that each vote type in V' either approves of no candidates (in which case we won't even display it; it is of no help toward meeting our control goal) or, necessarily, is an interval (although perhaps a degenerate one that contains exactly one candidate) with respect to L. Since we're handling the succinct case, votes of the same vote type are in the figure listed together along with their multiplicity to their right. We do this because in the input to the problem votes come in by type and with a listed-in-binary multiplicity, in the succinct case.

The number of vote types is $O(\|C\|^2)$, since each vote type, except for disapproving of everyone, is defined by its left and right endpoints. And, even more to the point, the number of vote types is trivially at most linear in the size of the input, since each vote type cast by at least one voter has to appear in the input. So the number of vote types certainly is not so large as to cause any challenges (about having enough time to consider the vote types occurring in V') regarding remaining within polynomial time.

Now, at first, things might seem worrisome. How can we decide which votes to add, especially between "incomparable" vote types such as, in our figure, the vote type with multiplicity 2 and the vote type with multiplicity 7? Those two vote types each help p and some other candidates. However, the sets of other candidates helped are incomparable—neither is a subset of the other.

We handle this worry by giving a "smart greedy" algorithm that only makes moves that are clearly at least as safe as any other move, e.g., that will never close down the last path to success if any path to success exists.

Let us look again at the figure. Note that although there are many candidates, we can ignore all candidates except p herself and the candidates that are shaded—l_6, l_3, l_2, r_3, r_6, and r_{10}. Why? Well, in our algorithm, we will never add any vote from V' that does not approve of p. In fact, let us from now on consider V' to be redefined to have removed from it every vote that does not approve of p. So in the figure, the V' vote types shown with multiplicity 5 and 9 go, as do any votes inside "etc." that do not approve of p. Now, since we will never add a vote from V' that does not approve of p (indeed, we redefined V' to drop such votes), and since each added vote will be an interval that includes p, notice that, for example, if we add some votes from V' that cause p to have (in total, between the original votes from V and added votes from V') strictly more approvals than r_3, then p necessarily will have strictly more approvals than r_1 and r_2, since each will have received from the added votes from V' no more additional approvals than p received from those added votes. By the same reasoning, if by adding votes from (our redefined) V' we ensure that p has strictly more approvals than r_6, clearly p will have strictly more approvals than r_4 and r_5. And the same argument applies to the case of r_{10} and, in mirror image, the shaded left-side bars. So these are the only "dangerous" contenders that we have to worry about regarding p. This observation will guide our choice of which votes to add and which candidates to ignore.

The reader may worry that the "by the same reasoning" statement four sentences ago is not a valid claim. In particular, one may worry that since r_5 starts above p in score, we may by adding votes that approve of both p and r_5 (but not r_6) end up with p having strictly more approvals than r_6 and yet with r_5 having more approvals than p, contrary to what our sentence above is asserting. The reason this reasonable worry is not actually a problem is that we will, as will be made clearer later in the proof, use a particular step-by-step process. The process we follow will, before even thinking of adding any votes approving r_5, first have (if there is any path to ensuring this; if not, control is impossible) ensured *via adding votes that give approval to p but not to* r_3 *(and thus not to* r_5*)* that p has more approvals than r_3. But note that after doing that, our "by the same reasoning" does cleanly apply to the case of r_4, r_5, and r_6. This point is not identical to, but rather works hand-in-hand with, the point of the next paragraph.

Before going on, let us mention why we two paragraphs ago did one step of what will be a step-by-step process, rather than simply trying to handle everything in one sweeping action. In particular, one might ask why we did not simply say that if p by adding votes from (our redefined) V' could be made to defeat r_{10} then it necessarily would defeat all of r_1 through r_9. But that claim in fact does not hold. Consider the case of adding 6 of the vote type having multiplicity 7 in the figure. Doing so boosts p to 8 approvals, passing r_{10}'s 7 approvals. But in the process, r_3 was boosted to 9, and so p does not beat r_3. That is, the argument used two paragraphs before the present one worked because the candidates we argued we could ignore were strictly less approved, currently, than the "dangerous" voter we left in that outflanked them. Just to be utterly specific, the first dangerous candidate to the right of p will be the leftmost candidate to the right of p that from V gets at least as many approvals as p (if we were in the nonunique-winner model, we would have said "strictly more approvals than p"). In the figure, that is r_3. And the next dangerous candidate is whichever candidate is the next candidate to the right of that candidate that from V has strictly more approvals than that candidate. In our figure, that is r_6. And so on. And analogously on the left.

So, now that we've set out who the dangerous rivals are, let us see if there is a choice of control action that can let us beat them. In this proof, we will show just how to beat the very first of them, and then will mention how to continue the process to (attempt to) defeat the rest of them. Crucially, the path we will take to defeat our first rival will be carefully chosen so that it is the safest possible path. That is, if there is any overall successful control action, then there will be one consistent with the actions we take to beat the first rival. And the iterative repeating of the scheme we outline in this proof will in fact then find such actions.

So let us focus on the first dangerous rival. In our figure, that would be r_3. (Note: If there are no dangerous rivals, then p is already a unique winner, and we're done. If the only dangerous rivals we have are on the left, then mirror image the universe, i.e., reverse L, and now all our dangerous rivals are on the right.) Now, to become the unique winner, p must strictly beat r_3. But note that among all the votes in V', the only ones that can possibly make that happen are (keeping in mind that we consider V' to now have in it only votes that approve of p) *those votes in* V' *whose right endpoint falls in the half-open interval* $[p, r_3)$. Call this set B. These are the only votes in V' that let p gain approvals relative to r_3. So, in any control action at all that causes p to strictly beat (and we mention that if we were in the nonunique-winner model, we'd here say "tie or beat") r_3, we must put enough votes from that collection, B, in to make that happen. So our only question is which of these votes to add, if there even are enough of them to add to lead to success. We do so as follows. We will add votes from among these *starting first with the rightmost left endpoint*. This is a perfectly safe strategy, as that choice helps the same or strictly fewer "left" dangerous candidates than would choosing a vote with a left endpoint that goes more to the left. As to the worry that by within B looking only at the left endpoint we may favor some *right-side* candidates that we might have otherwise slighted, that is so, but it is not a worry. We indeed are treating all endpoints between p and just before r_3 as a big blurred class. However, the differences among those endpoints affect just the (necessarily nondangerous) candidates in-between, namely, in our figure, candidates r_1 and r_2, and we already argued that by handling r_3 we will handle both of those. So giving extra approvals to one or both of those (that we could have avoided by making another choice within B) is not an issue. In short we've intentionally smeared our right-end finickiness because it doesn't matter, but left-end details matter a lot. So, in our figure, we need p to gain two approvals relative to r_3. Using the scheme/ordering just mentioned, we will among the votes in B keep eating through votes until we either achieve this or until we've hit the control problem's limit on the total number of votes we are allowed to add from V'. In this case, in our figure's example, the votes belonging to B (let us assume that the "etc." members of V' contain no members of B) are those votes contained inside the types $[l_4, p]$, $[l_3, r_1]$, and $[l_1, r_2]$.

The first vote type to draw on among these is (not withstanding that doing so helps r_2, unlike the case for the other two members of B) $[l_1, r_2]$, as it has the rightmost left endpoint. That vote type is of multiplicity 1, so we take that one vote. The next vote type to draw on is $[l_3, r_1]$. Even though that vote type is of multiplicity 2, p only needs one more approval to beat r_3, so we add to our election just one of the two of the votes of type $[l_3, r_1]$.

And so we have defeated our first dangerous rival, and in the safest possible way—a way that preserves a path to success if any exists.

The completion of the algorithm is basically to do this iteratively as long as candidates remain that, relative to the current state of the votes, are dangerous. So, after we remove a rival, we can consider all the added votes as if they are part of V now, and can redraw the figure, and can move forward, and do all the above against the next dangerous candidate on the right (of course, decreasing the limit on how many votes we are allowed to add by the number we just added), or if we run out of candidates on the right, we can mirror image L and start eating through any remaining dangerous candidates. (Regarding who the dangerous candidates are, the recomputation just mentioned is natural. But we mention in passing that the set of right-side dangerous candidates we will deal with in the algorithm is in fact the same as the set initially identified—the picture updating won't change it. However, while handling those candidates, it is possible that some of the additional approvals given to p will help it defeat some of the original left-side rivals, as a free side effect.) If we achieve our control goal before burning through the k allowed additional votes from V' (recall that k is part of the problem's input) then we have achieved our control goal, and otherwise it holds that our control goal cannot be achieved.

Let us now turn to the case of deleting voters. This case can be handled essentially symmetrically to the adding-voters case, except that we now have a very different choice of which voters to focus on and how to sort them in order to decide which to delete. So we merely highlight the differences between the two cases.

In the deleting-voters case, a picture showing our initial state would show just the bar chart of Fig. 2, i.e., the number of approvals from V each candidate gets initially. After deletion, the deleted votes would show up below the bar chart, each again as an interval corresponding to the candidates approved of by this voter. However, since it would be pointless to delete a voter approving of p, all such intervals do not include p.

Now, we try to make p strictly beat the next dangerous rival to the right, starting with r_3 in the example shown in Fig. 2—ignoring, of course, the intervals indicating the votes in V' and their multiplicities, as there is no V' in the deleting-voters case. To do so, we now consider all the (remaining, i.e., not yet deleted in the current iteration of our algorithm) votes with intervals whose left endpoint falls in the half-open interval $(p, r_3]$ (in the first iteration in our example). Now we sort these votes *from the rightmost right endpoint to the leftmost right endpoint*. As in the adding-voters case (except there we added votes), we now delete, following the sorted order, just enough votes so that p has one more approval than r_3. If that is impossible, we cannot control. But if it is possible, we've succeeded in making p defeat r_3 in the safest possible way. Recomputing the figure (where, in the redrawn picture, updating the number of the candidates' approvals implies that r_3 no longer is a dangerous candidate for p), we try to keep defeating p's remaining dangerous rivals. When we run out of dangerous candidates to the right of p, we again mirror image the universe by reversing L and try to make p defeat any remaining dangerous candidates. Eventually, this process ends when we either succeed in making p the unique winner by defeating all of p's dangerous rivals, or we exceed the deletion limit k before reaching this goal. All details not discussed here are obvious from the adding-voters proof. □

We turn from approval voting to plurality voting. Plurality voting's constructive control complexity was studied in detail in the seminal control paper of Bartholdi et al. [6]. The destructive cases were added by Hemaspaandra et al. [33] (see [24] for the limited adding of candidates cases). For plurality, the case-by-case situation is close to the opposite of that for approval. Regarding all the adding/deleting control cases defined in this paper, the four voter cases are vulnerable but all six candidate cases are resistant. However, our results here again are in keeping with our paper's theme: all six of these cases become vulnerable for single-peaked societies.

Theorem 3.2. *For the single-peaked case, plurality voting is vulnerable to constructive and destructive control by adding candidates, by adding unlimited candidates, and by deleting candidates, and these results hold in both the unique-winner model and the nonunique-winner model.*

This result holds in our settled model in which the linear order specifying the society's order on the candidates is part of the input, and also holds in the model in which the linear order is not part of the input but rather the question is whether there exists any valid linear order relative to which there is a way of achieving our control goal.

Our proofs of the different parts of this theorem vary greatly in approach. One approach that is particularly useful is dynamic programming. We defer the proof of Theorem 3.2 to Section 3.2.

For two very important voting systems, plurality and approval, we have seen that in every single adding/deleting case where they are known to have NP-hardness complexity shields, the complexity shield evaporates for societies with single-peaked preferences. In the coming manipulation section we will also see a number of cases where NP-hardness shields melt away for single-peaked societies. However, we will also see some cases—and earlier such a case was found by Walsh [43]—where existing NP-hardness shields remain in place even if one adds the restriction to a single-peaked society.

The previous paragraph raises the following natural question: given that restricting to single-peaked preferences can sometimes remove complexity shields, e.g., Theorem 3.1, and can sometimes leave them in place, e.g., Theorem 4.3, can

restricting to single-peaked preferences ever erect a complexity shield? It would be very tempting to hastily state that such behavior is impossible. After all the single-peaked case, if anything, thins out the flood of possible control-actions/manipulations. But we're talking about complexity here, and fewer options doesn't always mean a less complex problem. In particular, one of those manipulation/control-action options that single-peakedness takes off the table might have been a single, sure-fire path to victory under some election system. In fact, using precisely that approach, and a few other tricks, we have built an artificial election system for which unweighted constructive manipulation by size-3 coalitions is in P in the general case but is NP-complete for the single-peaked case. The system's votes are approval vectors. The system is highly unnatural and would never be considered for real-world use. However, its goal is merely to show that restricting to single-peakedness can, perhaps surprisingly, raise complexity. We conjecture that this strange behavior will never be seen for any existing, natural election system. We summarize this paragraph in the following theorem.

Theorem 3.3. *There exists an election system $\mathcal{E}$, whose votes are approval vectors, for which constructive size-3-coalition unweighted manipulation is in* P *for the general case but is* NP*-complete in the single-peaked model.*

This result holds both in our settled model where L, the society's ordering of the candidates, is part of the input and in the model where L is not given and the question is whether there is any societal ordering of the candidates that allows the manipulators to succeed.

See Footnote 8 of the technical report version [26] for a discussion of whether the *election system* itself should be allowed, in the single-peaked case, to have its actions depend on the input permutation relative to which V is single-peaked. Our construction and proof hold regardless of which approach one takes to this.

Proof. Let $C = \{c_1, \ldots, c_m\}$ be a set of candidates that is lexicographically ordered. That is, c_1 is the smallest candidate in this ordering. Note that this lexicographical ordering is not necessarily related to the society's ordering in single-peaked electorates. Let V be a collection of approval vectors over C. Given C and V, election system $\mathcal{E}$ works as follows. If $\|C\| \leq 2$, all candidates win. Otherwise (i.e., if $\|C\| \geq 3$) we have two possibilities:

1. If there exists no permutation making the given votes single-peaked (which, by Theorem 2.1, can be tested in polynomial time) then all candidates win. This feature of the system allows a manipulating coalition to in the general (single-peakedness not required) case cast votes precluding single-peakedness, an attack the single-peaked case does not allow.
2. Otherwise (i.e., if there is some permutation witnessing single-peakedness of the votes), if the lexicographically smallest candidate, c_1, when viewed as encoding a boolean formula encodes a satisfying one and the lexicographically smallest approval vector is all zeros, then all candidates win, else all candidates lose.

For the general case, the constructive size-3-coalition unweighted manipulation problem is in P: always have the three manipulators cast the following votes, with respect to the lexicographic ordering of $C = \{c_1, \ldots, c_m\}$:

$$(1, 1, 0, 0, \ldots, 0)$$
$$(0, 1, 1, 0, \ldots, 0)$$
$$(1, 0, 1, 0, \ldots, 0)$$

ensuring that the electorate is not single-peaked, and so all candidates win, including the distinguished candidate.

For the single-peaked case, however, the constructive size-3-coalition unweighted manipulation problem is NP-complete. The problem is clearly in NP. To show NP-hardness we reduce the well-known NP-complete boolean formula satisfiability problem to it. Given a boolean formula φ, encoded as a string in $\{0, 1\}^*$, map it to the instance (C, V, φ) of the constructive size-3-coalition unweighted manipulation problem, where $C = \{\varphi, 1^{|\varphi|+1}, 1^{|\varphi|+2}\}$, V consists of one nonmanipulator who approves of all candidates and three manipulators, and where φ is the distinguished candidate the manipulators want to make win. We claim that the manipulators can succeed in doing so exactly if φ is satisfiable. Indeed, if φ is satisfiable then φ can be made a winner by one of the manipulators casting the vote $(0, 0, 0)$. On the other hand, if φ is not satisfiable and the electorate is restricted to being single-peaked, then φ cannot win, no matter which votes the manipulators cast. □

3.2. The deferred proof of Theorem 3.2

In this section we provide, as a series of lemmas that cover appropriate cases, our proof of Theorem 3.2. One of the reasons why candidate-control problems for plurality are hard in the general case (i.e., when the voters are not restricted to be single-peaked) is that adding or deleting even a single candidate can affect the scores of all the other candidates. However, in the single-peaked case adding or deleting a candidate can affect at most two other candidates. This observation, formalized in the next lemma, is at the heart of the vulnerability proofs of this section.

Lemma 3.4. *Let (C, V) be an election where $C = \{c_1, \ldots, c_m\}$ is a set of candidates, V is a collection of voters whose preferences are single-peaked with respect to a linear order L, and where $c_1 \, L \, c_2 \, L \cdots L \, c_m$. Within plurality, if $m \geq 2$ then*

1. $score_{(C,V)}(c_1) = score_{(\{c_1,c_2\},V)}(c_1)$,
2. *for each* i, $2 \leq i \leq m-1$, $score_{(C,V)}(c_i) = score_{(\{c_{i-1},c_i,c_{i+1}\},V)}(c_i)$, *and*
3. $score_{(C,V)}(c_m) = score_{(\{c_{m-1},c_m\},V)}(c_m)$.

Proof. Let (C, V), L, and m be as in the statement of the lemma. We prove the second case, $2 \leq i \leq m-1$. The proofs of the boundary cases are analogous. Fix an integer i, $2 \leq i \leq m-1$. Each voter v in C that ranks c_i as her top choice clearly still prefers c_i to both c_{i-1} and c_{i+1} when limited to the choice between c_i, c_{i-1}, and c_{i+1}. On the other hand, consider a voter v in V who ranks some candidate c_j, $c_i \neq c_j$, as her most preferred candidate. If $j \leq i-1$ then, keeping in mind the single-peakedness of the preferences, clearly this voter prefers c_{i-1} to c_i. If $j \geq i+1$ then, for the same reason, this voter prefers c_{i+1} to c_i. In either case, this voter does not give her point to c_i. As a result, $score_{(C,V)}(c_i) = score_{(\{c_{i-1},c_i,c_{i+1}\},V)}(c_i)$. □

Using Lemma 3.4, we show that in the single-peaked case plurality is vulnerable to constructive control by adding candidates.

Lemma 3.5. *For the single-peaked case, plurality voting is vulnerable to constructive control by adding candidates both in the unique-winner model and in the nonunique-winner model.*

Proof. We give a polynomial-time algorithm for the single-peaked variant of the constructive control by adding candidates problem. We focus on the unique-winner case. However, the proof can easily be adapted to the nonunique-winner model, and we will be pinpointing the necessary changes throughout the proof. As per this problem's definition in Section 2, our input is (C, V, p, k, A) and a linear order L over $C \cup A$,[1] where $C = \{p, c_1, \ldots, c_{m'}\}$, $A = \{a_1, \ldots, a_{m''}\}$, k is a nonnegative integer, and the preferences in V are over $C \cup A$. After checking that the voters in V are single-peaked with respect to L, we ask whether it is possible to find a subset A' of A such that (a) $\|A'\| \leq k$ and (b) for each $c \in C \cup A'$, $c \neq p$, it holds that $score_{(C\cup A',V)}(c) < score_{(C\cup A',V)}(p)$ ($score_{(C\cup A',V)}(c) \leq score_{(C\cup A',V)}(p)$ in the nonunique-winner case). Let $D = C \cup A$ and let us rename the candidates so that $D = \{d_1, \ldots, d_m\}$, where $m = 1 + m' + m''$ and $d_1 \, L \, d_2 \, L \cdots L \, d_m$. Without loss of generality we can assume that p is neither d_1 nor d_m. If this p were d_1 or d_m, we could add to C two dummy candidates, d' and d'', that each voter ranks as the least desirable ones and such that $d' \, L \, d_1 \, L \cdots L \, d_m \, L \, d''$.

Before we describe the algorithm formally, let us explain it intuitively. Let us assume that our instance is a "yes" instance and that A' is a subset of A witnessing this fact. There must be two candidates in $C \cup A'$, d_ℓ and d_r, that are direct neighbors of p with respect to L restricted to $C \cup A'$, i.e., $d_\ell \, L \, p \, L \, d_r$ and there is no $d_i \in C \cup A'$ such that $d_i \neq p$ and $d_\ell \, L \, d_i \, L \, d_r$. By Lemma 3.4, the score of p is "fixed" by d_ℓ and d_r in the sense that $score_{(C\cup A',V)}(p) = score_{(\{d_\ell,p,d_r\},V)}(p)$. Also, given such d_ℓ and d_r we can view $C \cup A'$ as partitioned into a left part, $C_\ell \cup A'_\ell$, containing d_ℓ and the candidates preceding her, a right part, $C_r \cup A'_r$, containing d_r and the candidates succeeding her, and a middle part, containing p. The candidates in the left part and the candidates in the right part all have scores lower than p.

The idea of the algorithm is to try all possible pairs of candidates d_ℓ and d_r that can be direct neighbors of p and to test, for each such pair, whether it is possible to add candidates so that (a) each of the candidates in the left part has a score lower than p, (b) each of the candidates in the right part has a score lower than p, and (c) the number of candidates added is at most k. This approach is formalized in the algorithm below.

1. For each pair of candidates, d_ℓ and d_r, such that
   (a) $d_\ell \, L \, p \, L \, d_r$ and
   (b) there is no i, $1 \leq i \leq m'$, such that $d_\ell \, L \, c_i \, L \, d_r$,
   execute Steps 2 through 5 below. If after trying all pairs (d_ℓ, d_r) we have not accepted, then reject.
2. (a) Set $b' = score_{(\{d_\ell,p,d_r\},V)}(p) - 1$.
   (b) Set $C_\ell = \{c_i | c_i \, L \, d_\ell\} \cup \{d_\ell\}$ and $C_r = \{c_i | d_r \, L \, c_i\} \cup \{d_r\}$. (Note that we put d_ℓ in C_ℓ and d_r in C_r even if either or both of them belong to A.)
   (c) Set $A_\ell = \{a_i | a_i \, L \, d_\ell\}$ and $A_r = \{a_i | d_r \, L \, a_i\}$.
   (d) V_ℓ is the collection of those voters in V who, among the candidates in $C_\ell \cup C_r \cup A_\ell \cup A_r \cup \{p\}$, rank first either d_ℓ or a candidate d_i such that $d_i \, L \, d_\ell$. V_r is the collection of those voters in V who, among the candidates in $C_\ell \cup C_r \cup A_\ell \cup A_r \cup \{p\}$, rank first either d_r or a candidate d_i such that $d_r \, L \, d_i$.

[1] If we are in the model in which no linear societal order is given simply use, for example, the algorithm of Escoffier et al. [22] to find one relative to which our input is single-peaked. If none exists, state that and terminate. If the algorithm of [22] gives us some valid order, it is not hard to see that the algorithm we are about to describe in this proof will find a successful control action exactly if there is any valid societal linear order relative to which a successful control action exists. In the remaining proofs of Section 3.2, we will handle only the case that the societal order witnessing single-peakedness is given in our polynomial-time algorithms showing vulnerability claims. Note, however, that this footnote's argument handles the case of such a societal order not being given analogously in each such case.

Note that if B_ℓ is some subset of A_ℓ and B_r is some subset of A_r then in election $(C \cup B_\ell \cup B_r \cup \{d_\ell, d_r\}, V)$ it holds that (a) each voter in V_ℓ ranks first some candidate in $C_\ell \cup B_\ell \cup \{d_\ell\}$, (b) each voter in V_r ranks first some candidate in $C_r \cup B_r \cup \{d_r\}$, and (c) all the remaining voters rank p first.

3. Find the smallest natural number, call it β_ℓ, such that there exists a set B_ℓ, $B_\ell \subseteq A_\ell$, $\|B_\ell\| = \beta_\ell$, such that for each $c \in C_\ell \cup B_\ell$ it holds that $score_{(C_\ell \cup B_\ell, V_\ell)}(c) \leq b'$. If β_ℓ does not exist, then discard the current pair (d_ℓ, d_r) and try the next one.
4. Find the smallest natural number, call it β_r, such that there exists a set B_r, $B_r \subseteq A_r$, $\|B_r\| = \beta_r$, such that for each $c \in C_r \cup B_r$ it holds that $score_{(C_r \cup B_r, V_r)}(c) \leq b'$. If β_r does not exist, then discard the current pair (d_ℓ, d_r) and try the next one.
5. If $\beta_\ell + \beta_r + \|A \cap \{d_\ell, d_r\}\| \leq k$ then accept.

This algorithm, and statement and proof of Lemma 3.7, are written to show vulnerability. However, at the end of Section 2, we promised that all our vulnerability results would also hold for the stronger case of certifiable vulnerability. That latter claim requires us to in polynomial time not just test whether a successful manipulative action exists, but to produce it if it does. In almost all this paper's proofs, it is implicitly clear how to produce the successful action. However, for the current proof and that of Lemma 3.7, it is less clear, and so we now briefly discuss why certifiable vulnerability holds here. Suppose the condition in the first line of Step 5 holds. Let B_ℓ and B_r be sets witnessing the successful β_ℓ and β_r values. Then the successful action is to use, as our set A', $B_\ell \cup B_r \cup (\{d_\ell, d_r\} \cap A)$. To support this, the dynamic programming proof we'll give for Lemma 3.7 must be altered to not just find a integer value, but to give the set witnessing that value. Briefly put, one can indeed alter it to do that: using standard techniques for reconstructing solutions of dynamic programming algorithms we modify the forthcoming proof of Lemma 3.7, so that it in polynomial time even produces the set A' (of Definition 3.6) whose cardinality its dynamic programming algorithm is computing.

It is easy to see that the above algorithm is correct and that it also works in the nonunique-winner model if we replace Step 2a with $b' = score_{(\{d_\ell, p, d_r\}, V)}(p)$. We will now show that the algorithm runs in polynomial time. There are at most quadratically many (with respect to $m = \|D\|$) pairs (d_ℓ, d_r) to try, so each of the above steps will be executed at most polynomially often. It remains to show that Steps 3 and 4 can each be executed in polynomial time. To this end, we define the following functional problem.

Definition 3.6. In the DemoteByAddingCandidates problem we are given as input disjoint candidate sets C and A, vote collection V of linear orders over $C \cup A$, nonnegative integer b, and linear order L over $C \cup A$, and the votes in V are single-peaked with respect to L. We ask what the size is of a smallest set $A' \subseteq A$ such that for each candidate $c \in C \cup A'$ it holds that $score_{(C \cup A', V)}(c) \leq b$. If such a set A' does not exist then, by convention, the answer is ∞.

Computing β_ℓ (β_r) in Step 3 (Step 4) simply requires solving an instance of DemoteByAddingCandidates with input values $C_\ell, A_\ell, V_\ell, b'$, and L restricted to the appropriate subset of candidates (C_r, A_r, V_r, b', and L restricted to the appropriate subset of candidates). The following lemma establishes that this can be done efficiently.

Lemma 3.7. *DemoteByAddingCandidates is computable in polynomial time.*

Proof. We give a polynomial-time algorithm for the DemoteByAddingCandidates problem. Our input consists of disjoint candidate sets C and A, vote collection V of linear orders over $C \cup A$, a nonnegative integer b, and a linear order L over $C \cup A$, where $C = \{c_1, \ldots, c_{m'}\}$, $A = \{a_1, \ldots, a_{m''}\}$, and voters in V have single-peaked preferences with respect to the order L. Note that within the proof of this lemma, C, A, V, b, and L refer to the input to the DemoteByAddingCandidates problem, not to the input to the Control by Adding Candidates problem.

Let $D = C \cup A$ and let us rename the candidates so that $D = \{d_1, \ldots, d_m\}$, where $m = m' + m''$ and $d_1 \, L \, d_2 \, L \cdots L \, d_m$. Without loss of generality, we assume that $d_1 \in C$ and that $m \geq 3$. (If either of these did not hold, we could add to C an appropriate number of dummy candidates, each ranked by all the voters as less desirable than any of the candidates in $C \cup A$. Extending L so that for each added dummy candidate d it holds that $d \, L \, d_1$ and the voters are single-peaked with respect to this extended L is easy.) Let i, j, and k be three positive integers such that $1 \leq i < j < k \leq m$. We define

1. $s(i, j, k) = score_{(\{d_i, d_j, d_k\}, V)}(d_j)$,
2. $s(0, j, k) = score_{(\{d_j, d_k\}, V)}(d_j)$, and
3. $s(i, j, m+1) = score_{(\{d_i, d_j\}, V)}(d_j)$.

Note that, by Lemma 3.4, if B is a subset of A such that (a) $\{d_i, d_j, d_k\} \subseteq C \cup B$ and (b) there is no candidate d_l ($l \neq j$) such that $d_i \, L \, d_l \, L \, d_k$, then $s(i, j, k) = score_{(C \cup B, V)}(d_j)$.

Let i and j be two positive integers such that $1 \leq i < j \leq m$. By $A(i, j)$ we mean a family of subsets of A such that a set B, $B \subseteq A$, belongs to $A(i, j)$ if and only if (a) $d_i \in C \cup B$, (b) $d_j \in C \cup B$, and (c) $(C \cup B) \cap \{d_{i+1}, \ldots, d_{j-1}\} = \emptyset$.

We now define a key value for our algorithm. Let i and j be two positive integers such that $1 \leq i < j \leq m$. We define

$$f(i,j) = \min_{B \in A(i,j)} \{\|B\| | (\forall d_\ell \in C \cup B, 1 \leq \ell \leq i)[score_{(C \cup B, V)}(c_\ell) \leq b]\}.$$

We adopt the convention that $\min \emptyset$ is ∞. It is easy to verify that the algorithm's output should be

$$\min_{1 \leq i < j \leq m} \{f(i,j) | \{d_{j+1}, \ldots, d_m\} \subseteq A \wedge s(i,j,m+1) \leq b\}.$$

It remains to show that we can compute the values $f(i,j)$, $1 \leq i < j \leq m$, in polynomial time. The rest of the proof is devoted to this task.

We compute the values $f(i,j)$ using dynamic programming. Let j be a positive integer $1 < j \leq m$. It is easy to see that

$$f(1,j) = \begin{cases} \infty & \text{if there is some } d_i \in C \text{ such that } 1 < i < j, \\ \infty & \text{if the above does not hold but } s(0,1,j) > b, \\ \|A \cap \{d_1, d_j\}\| & \text{otherwise.} \end{cases}$$

Given these initial values,[2] we can compute arbitrary values $f(j,k)$, $1 < j < k \leq m$, using the following facts. Let us fix j and k such that $1 < j < k \leq m$. If there is a $d_l \in C$ such that $j < l < k$ then $f(j,k) = \infty$ (because $A(j,k)$ is empty). Otherwise

$$f(j,k) = \min_{1 \leq i < j} \{f(i,j) + \chi_A(d_k) | s(i,j,k) \leq b\},$$

where χ_A is the characteristic function of A, i.e., $\chi_A(d_k)$ is 1 if $d_k \in A$ and is 0 otherwise. It is easy to see that using standard dynamic-programming techniques and the above equations we can compute $f(i,j)$ for arbitrary positive integers i and j, $1 \leq i < j \leq m$, in polynomial time.

This completes the proof of Lemma 3.7. □

This completes the proof that for the case of single-peaked voters, plurality is vulnerable to constructive control by adding candidates.

This completes the proof of Lemma 3.5. □

As a corollary to Lemma 3.5, we obtain the corresponding result for the case of adding an unlimited number of candidates. This is because for each unlimited case it is sufficient to use the algorithm for the corresponding limited case with the limit on the number of candidates that can be added set to the number of spoiler candidates.

Corollary 3.8. *For the single-peaked case, plurality voting is vulnerable to constructive control by adding an unlimited number of candidates in both the unique-winner model and the nonunique-winner model.*

In the following proofs, both in the destructive adding-candidates cases and in the deleting-candidates cases, we will need Lemma 3.4 and one additional tool. That tool is the notion of a neighborhood of a candidate c in an order L witnessing single-peakedness of an electorate.

Definition 3.9. Let (C,V) be an election and let L be an order such that the voters in V are single-peaked with respect to L. Let c be a candidate in C. We rename the candidates in C such that $C = \{b_{m'}, \ldots, b_2, b_1, c, d_1, d_2, \ldots, d_{m''}\}$ and $b_{m'} L \cdots L\, b_2 L\, b_1 L\, c L\, d_1 L\, d_2 L \cdots L\, d_{m''}$. For each two positive integers i and j we set $D_{ij}(C,c) = \{b_1, \ldots, b_{\min(i,m')}\} \cup \{d_1, \ldots, d_{\min(j,m'')}\}$. We define the direct neighborhood of a candidate c to be the set $D(C,c) = \{D_{ij}(C,c) | 0 \leq i \leq m' \text{ and } 0 \leq j \leq m''\}$.

The notation of this definition, in essence, ties the order L to the candidate set C. We will sometimes wish to speak of, e.g., $D_{ij}(C',c)$ for some $C' \subseteq C$, $c \in C'$, and this notation allows us to naturally, implicitly speak of an order L' induced by C'.

Clearly, given a single-peaked election (C,V) and a candidate $c \in C$, $\|D(C,c)\|$ is polynomial in $\|C\|$. We now turn to the destructive adding-candidates cases in both the unique-winner model and the nonunique-winner model. The proof is much simpler than in the constructive cases.

Lemma 3.10. *For the single-peaked case, plurality voting is vulnerable to destructive control by adding candidates in both the unique-winner model and the nonunique-winner model.*

Proof. Our input is C, A, V, k, and L. $(C \cup A, V)$ is an election, where $C = \{d, c_1, \ldots, c_m\}$, $A = \{a_1, \ldots, a_{m'}\}$, and V is a collection of voters whose preference orders are single-peaked with respect to a given order L. The candidates from C are already registered and the candidates from A can be added (i.e., A is the spoiler candidate set). k is a nonnegative integer. We

[2] Note that here it is important that $d_1 \in C$. If this were not the case, we would also have to directly compute the values $f(i,j)$ for some i and j where $1 < i < j$.

now give a polynomial-time algorithm that tests whether there exists a set $A' \subseteq A$ such that (a) $\|A'\| \leq k$ and (b) d is not a unique winner of $(C \cup A', V)$.

The algorithm works as follows: if for some at-most-3-element subset A'' of A, d is not a unique winner of $(C \cup A'', V)$ then accept. And otherwise, reject. This algorithm clearly works in polynomial time. It remains to show that it is correct. If for each $A' \subseteq A$ it holds that d is a unique winner of $(C \cup A', V)$ then the above algorithm correctly rejects. On the other hand, let us assume that for some $A' \subseteq A$ it holds that d is not a unique winner of $(C \cup A', V)$ and let us fix one such A'. We claim that there is an at-most-3-element subset A'' of A such that d is not a unique winner of $(C \cup A'', V)$.

Let $B = D_{1,1}(C \cup A', d)$, that is, B contains the direct neighbors of d (among the candidates in $C \cup A'$, with respect to L). By definition, $\|B\| \leq 2$ and, from Lemma 3.4, $score_{(C\cup A',V)}(d) = score_{(B\cup\{d\},V)}(d)$. Since d is not a unique winner of $(C \cup A', V)$, there is some candidate $c \in C \cup A'$, $c \neq d$, such that $score_{(C\cup A',V)}(c) \geq score_{(C\cup A',V)}(d)$. Define $A'' = (B \cup \{c\}) \cap A$. Clearly, $\|A''\| \leq 3$ and, since B is a "radius-1" neighborhood of d, we have $score_{(C\cup A'',V)}(d) = score_{(B\cup\{d\},V)}(d)$. On the other hand, since $C \cup A'' \subseteq C \cup A'$ and c belongs to both sets, it holds that $score_{(C\cup A'',V)}(c) \geq score_{(C\cup A',V)}(c)$. It follows that in election $(C \cup A'', V)$ candidate c has a score at least as high as that of d and thus d is not a unique winner of $(C \cup A'', V)$.

It is easy to see that the same approach can be used to solve the nonunique-winner case. Our algorithm now tries all at-most-3-element subsets A'' of A and accepts if and only if for some such A'' it holds that d is not a winner of $(C \cup A'', V)$. The proof of correctness is analogous to the unique-winner case. □

As an easy corollary we obtain that in the single-peaked case plurality is also vulnerable to destructive control by adding unlimited candidates.

Corollary 3.11. *For the single-peaked case, plurality voting is vulnerable to destructive control by adding an unlimited number of candidates in both the unique-winner model and the nonunique-winner model.*

We now turn to the constructive and destructive deleting-candidates cases in both the unique-winner model and the nonunique-winner model. The main idea of our algorithms for the deleting-candidates cases is that it is sufficient to focus on deleting candidates adjacent to the preferred one (to increase her score) and then to delete those remaining candidates that still defeat the preferred one.

Lemma 3.12. *For the single-peaked case, plurality voting is vulnerable to constructive and destructive control by deleting candidates in both the unique-winner model and the nonunique-winner model.*

Proof. We give a polynomial-time algorithm for the single-peaked case of constructive control by deleting candidates for plurality. We focus on the unique-winner model but it is easy to see how to modify our algorithm to work in the nonunique-winner model.

Our algorithm's input contains an election (C, V), a preferred candidate $p \in C$, a nonnegative integer k (the number of candidates that we are allowed to delete), and an order L such that the voters in V are single-peaked with respect to L.

By Lemma 3.4, we see that the only possible scores that p can have after deleting some candidates are in the set $S = \{score_{(C-D,V)}(p) | D \in D(C, p)\}$ and that each of these scores can be obtained by deleting some subset $D \in D(C, p)$ of the candidates. In consequence, to ensure that p is a unique winner we first need to delete some subset $D \in D(C, p)$ of candidates and then—since it is impossible to decrease a candidate's score via deleting other candidates—delete all those candidates that still have more points than p. Of course, we do not know *a priori* which set D in $D(C, p)$ to delete, but since $\|D(C, p)\|$ is polynomially bounded in $\|C\|$, we can try all its members.

Formally, our algorithm works as follows:

1. For each $D \in D(C, p)$, execute Step 2. If after trying all D's we have not yet accepted, then reject.
2. (a) While there exists a candidate $c \in C - D$, $c \neq p$, such that $score_{(C-D,V)}(c) \geq score_{(C-D,V)}(p)$, add c to D.
   (b) If $\|D\| \leq k$ then accept.

Clearly, this algorithm works in polynomial time and, by the preceding discussion, it accepts if and only if it is possible to ensure p's victory via control by deleting candidates. For the nonunique-winner case, we simply need to change the inequality in Step 2a from "$\geq$" to "$>$."

In the destructive case we are given an election (C, V), a despised candidate $d \in C$, a nonnegative integer k (the number of candidates we can delete), and an order L such that the voters in V are single-peaked with respect to L. We assume that d is a unique plurality winner of (C, V); otherwise, we can immediately accept. It is sufficient to find a candidate c and a subset $D \subseteq C - \{d\}$ of candidates such that $\|D\| \leq k$ and $score_{(C-D,V)}(c) \geq score_{(C-D,V)}(d)$. As in the constructive case, by Lemma 3.4, it suffices to try each candidate $c \in C - \{d\}$ and each set $D \in D(C, c)$. That is, if there is a candidate $c \in C - \{d\}$ and a set $D \in D(C, c)$ such that (a) $\|D\| \leq k$, (b) $d \notin D$, and (c) $score_{(C-D,V)}(c) \geq score_{(C-D,V)}(d)$, then we accept. Otherwise we reject. Clearly, this algorithm works in polynomial time and, by Lemma 3.4 and the fact that deleting candidates does

not decrease the remaining candidates' scores, is correct. The algorithm can be modified in the obvious way to work for the nonunique-winner model. □

4. Manipulation

In this section we study constructive coalition weighted manipulation. Recall that in the single-peaked case the manipulators must cast votes that are consistent with the linear ordering of the candidates (which is part of the input in the single-peaked case) that defines the society's single-peakedness. However, all our "single-peaked case is in P" results in this section also hold in a model in which the linear order of the candidates is not part of the input, and we will comment on these cases in our proofs below.

This paper's theme is that single-peakedness removes many NP-hardness shields. For manipulation we support that theme with Theorem 4.1 (and its implications within Theorem 4.2) and Theorem 4.4. Note that in the general case, the election systems of parts 1 and 3 of Theorem 4.1 and the "$k_1 \geq 2 \wedge k_0 \geq 1$" cases of part 2 of this theorem are known to be NP-complete [16,32,41], and the remaining part 2 cases are easily seen to be in P [32].

Theorem 4.1. *For the single-peaked case, the constructive coalition weighted manipulation problem (in both the nonunique-winner model and the unique-winner model) for each of the following election systems is in* P*:*

1. *The scoring protocol* $\alpha = (2, 1, 0)$*, i.e., 3-candidate Borda elections.*
2. *Each scoring protocol* $\alpha = (\overbrace{1, \ldots, 1}^{k_1}, \overbrace{0, \ldots, 0}^{k_0})$, $k_1 \geq k_0$*. (This includes a variety of* ℓ*-veto and* ℓ'*-approval protocols, e.g., the 3-veto cases for* $m \geq 6$ *candidates in Theorem 4.2.)*
3. *Veto.*

All three of Theorem 4.1's cases hold both in our settled model where L, the society's ordering of the candidates, is part of the input and in the model where L is not given and the question is whether there is any societal ordering of the candidates that allows the manipulators to succeed.

Proof. For each of the election systems stated in the theorem, we will give or reference a polynomial-time algorithm solving the constructive coalition weighted manipulation problem for the single-peaked case. Each of these algorithms takes as input a list C of candidates, a list S of nonmanipulative voters each specified by a single-peaked preference over C (with respect to the given linear ordering L on C that defines single-peakedness) and an integer weight, a list of the weights of the voters in our manipulating coalition T, and the candidate $p \in C$ that the manipulators in T seek to make a winner (a unique winner). Note that all manipulators in T must also have single-peaked preferences with respect to L.

Part 1 is a special case of Theorem 4.4. A direct, simple proof of this part can be found in this paper's technical report version [26].

We now prove part 2. Let $\alpha = (\overbrace{1, \ldots, 1}^{k_1}, \overbrace{0, \ldots, 0}^{k_0})$, $k_1 \geq k_0$, be a scoring protocol. We will handle the cases $k_1 > k_0$ and $k_1 = k_0$ separately. As above, we will present algorithms for both the nonunique-winner model and the unique-winner model.

First, assume $k_1 > k_0$. In any election with $m = k_0 + k_1$ candidates, if the voters are single-peaked with respect to some linear order L,[3] then the middle candidate(s) (namely, the $\lceil m/2 \rceil$nd candidate in L if m is odd, and the $(m/2)$nd and the $(1 + m/2)$nd candidate in L if m is even) will always be among the winners. Thus given an instance of the constructive coalition weighted manipulation problem, our distinguished candidate p can be made a winner in $(C, S \cup T)$ if and only if p does not lose a point in S. So our algorithm checks whether p does not lose a point in S and accepts or rejects accordingly.

For the unique-winner case within scoring protocol $\alpha = (\overbrace{1, \ldots, 1}^{k_1}, \overbrace{0, \ldots, 0}^{k_0})$, $k_1 > k_0$, p can be made a unique winner if and only if

1. p does not lose a point in S, and
2. for all candidates c that are tied with p in S, it is possible to make c lose a point in T (while keeping T single-peaked with respect to L).

Since the number of candidates is fixed, our algorithm can easily check whether these two conditions hold and then accepts or rejects accordingly.

Now, assume $k_1 = k_0$. Given an instance of the constructive coalition weighted manipulation problem, it is enough to consider a linear order L (which witnesses single-peakedness; also, recall Footnote 3) that ranks at least $k = k_0 = k_1$

[3] If L is not part of the input, we simply try all possible societal orderings. This can be done in polynomial time, since there are only a constant number of candidates.

candidates before p and fewer than k candidates after p: $c_1 \, L \, c_2 \, L \, \cdots \, L \, c_\ell \, L \, p \, L \, c_{\ell+2} \, L \, \cdots \, L \, c_{2k}$ with $\ell \geq k$. For every candidate c_i, $\ell + 2 \leq i \leq 2k$, to the right of p, we have $score_S(p) \geq score_S(c_i)$, and for every candidate c_j, $k + 1 \leq j \leq \ell$, to the left of p and to the right of the middle, it holds that if p gets a point then so does c_j. It follows that p can be made a winner if and only if p is a winner in $(C, S \cup T)$ when every voter in T ranks the candidates from right to left by $c_{2k} > c_{2k-1} > \cdots > c_{\ell+2} > p > c_\ell > \cdots > c_1$.

For the unique-winner case within scoring protocol $\alpha = (\overbrace{1, \ldots, 1}^{k}, \overbrace{0, \ldots, 0}^{k})$, if p's right neighbor in L is tied with p in S (i.e., $score_S(c_{\ell+2}) = score_S(p)$), then set a voter of lowest weight in T to $p > c_\ell > c_{\ell-1} > \cdots > c_1 > c_{\ell+2} > \cdots > c_{2k}$ and set all other voters in T to rank the candidates from right to left by $c_{2k} > c_{2k-1} > \cdots > c_{\ell+2} > p > c_\ell > \cdots > c_1$. (Note that T is single-peaked with respect to L.) Our algorithm can easily check whether this makes p a unique winner in $(C, S \cup T)$, and if so it accepts, otherwise it rejects.

We now come to part 3, which regards veto elections, i.e., scoring protocols $\alpha = (1, \ldots, 1, 0)$ where the number of candidates varies. Similarly to the proof of part 2, this hinges on the following observation: given an instance of the constructive coalition weighted manipulation problem, where the votes are single-peaked with respect to the society's order L, p can be made a winner if and only if p is the only candidate or p is never last in S.

For the unique-winner case, note that since at most two candidates can be ranked last in single-peaked elections, unique winners within veto elections can only exist if $\|C\| \leq 3$. The result for $\|C\| \in \{2, 3\}$ follows from the previous part of the theorem, and the $\|C\| = 1$ case is trivial. □

We now come to an unusual case. For general votes (not limited to single-peaked societies), the constructive coalition weighted manipulation problem for 3-veto is in P for three or four candidates, and is NP-complete for five or more candidates [32]. Note that 3-veto is not meaningfully defined for two or fewer candidates. In contrast with the general case, for single-peaked votes 3-veto shows a remarkable behavior: moving from five to six candidates *lowers* the complexity of this manipulation problem. In some papers, authors state—e.g., page 9 of [16], in a context in which the claim holds perfectly well—that if one knows the number of candidates, if any, at which a system switches from easy to manipulate to hard to manipulate, then it is easy by adding dummy candidates to see that for all larger number of candidates the system remains hard. The following theorem should stand as a caution to take that view only if one has carefully built and checked a "dummy candidates" construction for one's specific case.

Theorem 4.2. *For the single-peaked case, the constructive coalition weighted manipulation problem (in both the unique-winner model and the nonunique-winner model) for m-candidate 3-veto elections is in* P *for* $m \in \{3, 4, 6, 7, 8, \ldots\}$ *and is resistant (indeed,* NP*-complete) for* $m = 5$.

This result holds both in our settled model where L, the society's ordering of the candidates, is part of the input and in the model where L is not given and the question is whether there is any societal ordering of the candidates that allows the manipulators to succeed.

Proof. Consider the following four cases.

$m = 3$ case. In this case, we are looking at the scoring protocol $(0, 0, 0)$. It is immediate that in this scoring protocol all candidates are always tied for winner, and so p can always be made a winner. For the same reason, p can never be a unique winner.

$m = 4$ case. In this case, we are looking at the scoring protocol $(1, 0, 0, 0)$. It is immediate that p can be made a winner (a unique winner) if and only if p is a winner (a unique winner) in the election where every manipulator ranks p first.

$m = 5$ case. In this case, we are looking at the scoring protocol $(1, 1, 0, 0, 0)$. It is immediate that the constructive coalition weighted manipulation problem is in NP, namely, guess single-peaked votes for the manipulators and verify that p is a winner of the election.

To show NP-hardness, we reduce from PARTITION. Given an instance of PARTITION, i.e., a set $\{k_1, \ldots, k_n\}$ of n distinct positive integers that sum to $2K$, construct the following instance of Constructive Coalition Weighted Manipulation: the set of candidates is $C = \{a, b, c, d, p\}$ with p being our distinguished candidate. The set of nonmanipulators S consists of two voters, each of weight K. One of the voters votes $c > a > p > b > d$ and the other voter votes $d > b > p > a > c$. Note that these two voters fix the society's order. We also have a set T of n manipulators. The weights of the manipulators are $k_1, k_2, \ldots, k_n$. We claim that there is a partition if and only if the manipulators can cast single-peaked votes that make p a winner.

First suppose that there exists a subset S' of $\{k_1, \ldots, k_n\}$ that sums to K. For every $i \in \{1, \ldots, n\}$, we set the weight k_i manipulator to $p > a > b > c > d$ if k_i in S' and to $p > b > a > c > d$ otherwise. In the resulting election, a, b, and p each score $2K$ points and c and d score K points. It follows that p is a winner of the election.

For the converse, suppose the voters in T cast votes that are single-peaked with respect to the societal order enforced by the above construction and that p is a winner of the election. Since for every single-peaked vote, if p gets a point then a or b gets a point, it follows that $score_T(p) \leq score_T(a) + score_T(b)$. Since p is a winner of the resulting election, $score_T(p) \geq K + score_T(a)$ and $score_T(p) \geq K + score_T(b)$. It follows that $score_T(p) = 2K$ and that $score_T(a) = score_T(b) = K$. But then the weights of the voters in T that give a point to a sum to K and so we have found a partition.

Changing the weights of the voters in S from K to $K-1$ handles the unique-winner case.
***m* ≥ 6 case.** This has already been proven as the special case $k_1 \geq k_0 = 3$ of part 2 of Theorem 4.1. □

We now present some cases that are known to be NP-hard in the general case (see [32]) and that we can prove remain hard even in the single-peaked case.

Theorem 4.3. *For the single-peaked case, the constructive coalition weighted manipulation problem (in both the unique-winner model and the nonunique-winner model) is resistant (indeed,* NP*-complete) for the following scoring protocols:*

1. $\alpha = (3, 1, 0)$.
2. $\alpha = (3, 2, 1, 0)$, *i.e., 4-candidate Borda elections.*

Both cases of Theorem 4.3 hold both in our settled model where L, the society's ordering of the candidates, is part of the input and in the model where L is not given and the question is whether there is any societal ordering of the candidates that allows the manipulators to succeed.

Part 1 of Theorem 4.3 is a special case of Theorem 4.4. A direct, simple proof of this part can be found in this paper's technical report version [26]. Part 2's proof, although more involved, is similar and for this too we refer the reader to the technical report.

We can extend the ideas behind the proofs of this section's theorems to obtain Theorem 4.4, a dichotomy result for 3-candidate scoring protocols in the single-peaked case. Supporting our paper's general theme that single-peakedness removes many NP-hardness shields, we mention that for the general (i.e., not required to be single-peaked) case, the analogous result for three-candidate scoring protocols $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, $\alpha_1 \geq \alpha_2 \geq \alpha_3$, is that the constructive coalition weighted manipulation problem (in both the unique-winner model and the nonunique-winner model) is resistant (indeed, NP-complete) when $\alpha_2 > \alpha_3$, and is in P otherwise [16,32,41]. So in the general case, all scoring protocols that are not in essence triviality or plurality are resistant.

Theorem 4.4. *Consider a 3-candidate scoring protocol, namely, $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, $\alpha_1 \geq \alpha_2 \geq \alpha_3$, $\alpha_1 \in \mathbb{N}$, $\alpha_2 \in \mathbb{N}$, $\alpha_3 \in \mathbb{N}$. For the single-peaked case, the constructive coalition weighted manipulation problem (in both the unique-winner model and the nonunique-winner model) is resistant (indeed,* NP*-complete) when $\alpha_1 - \alpha_3 > 2(\alpha_2 - \alpha_3) > 0$ and is in* P *otherwise.*

This result holds both in our settled model where L, the society's ordering of the candidates, is part of the input and in the model where L is not given and the question is whether there is any societal ordering of the candidates that allows the manipulators to succeed.

Proof. Consider a 3-candidate scoring protocol $\alpha = (\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_1 \in \mathbb{N}$, $\alpha_2 \in \mathbb{N}$, $\alpha_3 \in \mathbb{N}$, and $\alpha_1 \geq \alpha_2 \geq \alpha_3$. Without loss of generality, we may assume that $\alpha_3 = 0$. If α_3 were not 0, we could replace our scoring protocol with $(\alpha_1 - \alpha_3, \alpha_2 - \alpha_3, 0)$. Suppose that $\alpha_1 - \alpha_3 > 2(\alpha_2 - \alpha_3) > 0$, i.e., $\alpha_1 > 2\alpha_2 > 0$. We show that in this case the constructive coalition weighted manipulation problem is NP-complete. Membership in NP is immediate and to prove NP-hardness we reduce from PARTITION. Given a PARTITION instance, i.e., a set $\{k_1, \ldots, k_n\}$ of n distinct positive integers that sum to $2K$, we construct an instance of the constructive coalition weighted manipulation problem as follows. Let a, b, and p be the candidates in C, where p is the distinguished candidate the manipulators want to make win. There are two nonmanipulators in S, with preferences $a > p > b$ and $b > p > a$ and both having weight $(2\alpha_1 - \alpha_2)K$. There are n manipulators in T, where the ith manipulator has weight $(\alpha_1 - 2\alpha_2)k_i$, $1 \leq i \leq n$. Note that, due to $\alpha_1 > 2\alpha_2 > 0$, each voter has a positive integer weight. We fix the society's order L to be $a\,L\,p\,L\,b$. Note that L and its reverse are the only two orders consistent with the votes in S, so even in the model where L is not part of the input, we will still, in effect, be working with L as the society's order.

We claim that there exists a partition if and only if the manipulators can cast single-peaked votes (with respect to L) that make p a winner of $(C, S \cup T)$.

Suppose there is a partition, i.e., a set $A \subseteq I = \{1, 2, \ldots, n\}$ such that $\sum_{i \in A} k_i = \sum_{i \in I - A} k_i = K$. To make p win, set $(\alpha_1 - 2\alpha_2)K$ vote weight of T (say, each of the manipulators corresponding to A) to $p > a > b$ and the remaining manipulators to $p > b > a$. Note that T is single-peaked with respect to L. We have the following scores:

$$score_{S \cup T}(p) = 2\alpha_2(2\alpha_1 - \alpha_2)K + 2\alpha_1(\alpha_1 - 2\alpha_2)K = 2\left(\alpha_1^2 - \alpha_2^2\right)K,$$
$$score_{S \cup T}(a) = \alpha_1(2\alpha_1 - \alpha_2)K + \alpha_2(\alpha_1 - 2\alpha_2)K = 2\left(\alpha_1^2 - \alpha_2^2\right)K, \text{ and}$$
$$score_{S \cup T}(b) = \alpha_1(2\alpha_1 - \alpha_2)K + \alpha_2(\alpha_1 - 2\alpha_2)K = 2\left(\alpha_1^2 - \alpha_2^2\right)K.$$

So p is a winner of election $(C, S \cup T)$.

Conversely, suppose the votes in T can be cast such that p is a winner of $(C, S \cup T)$. Note that

$$\sum_{c \in C} score_{S \cup T}(c) = 6\left(\alpha_1^2 - \alpha_2^2\right) K.$$

So in order for p to be a winner, p's score in $(C, S \cup T)$ needs to be at least $2\left(\alpha_1^2 - \alpha_2^2\right) K$. This can happen only if p is ranked first by every voter in T. And if we set T's votes like that, p's score is exactly $2\left(\alpha_1^2 - \alpha_2^2\right) K$. In order for p to be a winner, a's score and b's score need to be $2\left(\alpha_1^2 - \alpha_2^2\right) K$ each. This means that the vote weight of the voters in T voting $p > a > b$ is equal to the vote weight of the voters in T voting $p > b > a$. Thus there exists a partition.

Since the reduction is polynomial-time computable, the constructive coalition weighted manipulation problem for the scoring protocol $\alpha = (\alpha_1, \alpha_2, 0)$ is NP-complete, provided that $\alpha_1 > 2\alpha_2 > 0$.

The unique-winner case can be handled by multiplying all weights in the above construction by a suitable constant (depending on α_1 and α_2) and then subtracting 1 from the weights in S. The idea is that subtracting 1 from the weights in S ensures that p can become the unique winner if there is a partition, and multiplying the weights, prior to the subtraction, ensures that the subtraction does not have any side effects that would invalidate the (analogue of the) above correctness argument in this modified reduction.

Now suppose $\alpha_1 \geq \alpha_2 \geq 0$ but it is *not* the case that $\alpha_1 > 2\alpha_2 > 0$. We show that in this case the constructive coalition weighted manipulation problem is in P. If $\alpha_2 = 0$ then our scoring protocol is equivalent to plurality for three candidates, and we accept if and only if p is a winner (a unique winner) when all the manipulators rank p first. Let us now focus on the case where $\alpha_2 > 0$, and so, by assumption, $\alpha_1 \leq 2\alpha_2$. Without loss of generality, it is enough to consider the following two orderings L witnessing the society's single-peakedness. (Again, if L is not part of the input, we simply try all societal orderings.) We will handle the nonunique-winner case and the unique-winner case in parallel.

Case 1: $a\,L\,b\,L\,p$. In this case, if the goal is to make p a winner (a unique winner), a best vote for all manipulators in T is clearly $p > b > a$.

Case 2: $a\,L\,p\,L\,b$. To be consistent with L, the only votes allowed in S are $p > a > b$, $p > b > a$, $a > p > b$, and $b > p > a$. Due to which votes are allowed in S, we have $2 \cdot score_S(p) \geq score_S(a) + score_S(b)$. It follows that $score_S(p) \geq score_S(a)$ or $score_S(p) \geq score_S(b)$. If we are in the nonunique-winner model, set all voters in T to $p > a > b$ if $score_S(p) \geq score_S(a)$, and to $p > b > a$ if $score_S(p) \geq score_S(b)$. If we are in the unique-winner model and $\alpha_1 > \alpha_2$, set all voters in T to $p > b > a$ if $score_S(a) \geq score_S(b)$ and to $p > a > b$ otherwise. The $\alpha_1 = \alpha_2$ case follows from part 2 of Theorem 4.1.

In both cases, we accept if the manipulation action described above was successful, i.e., made p a winner in the nonunique-winner case (a unique-winner in the unique-winner case) in election $(C, S \cup T)$, and we reject otherwise. □

Finally, we mention that despite the NP-hardness results of Theorems 4.3 and 4.4, in all these cases there are polynomial-time manipulation algorithms for the case when the candidate the coalition wants to win is either the top or the bottom candidate in society's input linear order on the candidates.

5. Related work

The paper that inspired our work is Walsh's "Uncertainty in Preference Elicitation and Aggregation" [43]. Among other things, in that paper he raises the issue of manipulation in single-peaked societies. Our paper follows his model of assuming society's linear ordering of the candidates is given and that manipulative voters must be single-peaked with respect to that ordering. However, our theme and his differ. His manipulation results present cases where single-peakedness leaves an NP-completeness shield intact. In particular, for both the constructive and the destructive cases, he shows that the coalition weighted manipulation problem for the single transferable vote election rule for three or more candidates remains NP-hard in the single-peaked case. Although our Theorem 4.3 follows this path of seeing where shields remain intact for single-peaked preferences, the central focus of our paper is that single-peaked preferences often remove complexity shields on manipulation and control. We mention in passing that for a different, noncontrol issue regarding the effect of single-peakedness—namely, looking at incomplete profiles and asking whether some/all the completions make a candidate a winner—Walsh's paper proves both P results and NP-completeness results. We are much indebted to his paper for raising and exploring the issue of manipulation for single-peaked electorates.

As mentioned in the main text, Bartholdi and Trick [7], Doignon and Falmagne [18], and Escoffier et al. [22] have provided efficient algorithms for testing single-peakedness and producing a valid candidate linear ordering, for the case when votes are linear orders. For the case of voting by approval vectors, this was achieved even earlier [11,29].

Since with single-peaked preferences there is always a Condorcet winner when the number of voters is odd, every Condorcet-consistent voting rule has an efficient winner-determination algorithm for the single-peaked case for odd numbers of voters [43].

Other work is more distant from our work but worth mentioning. Conitzer [14] has done an interesting, detailed study showing that, in the model where votes are linear orders, preferences in single-peaked societies can be quickly elicited via comparison queries ("Do you prefer candidate i to candidate j?"). He studies the case when the linear order of society is known and the case when it is not. We mention in passing that we have looked at the issue of preference elicitation in single-peaked societies (where the linear order is given) of approval vectors via approval queries ("Do you approve of candidate i?"). It is immediately obvious that single-peakedness gives no improvement for approval vectors and 1-approval vectors (approval vectors with exactly one 1; this is a vote type and should not be confused with "1-approval" as it would be used in scoring systems, where the actual vote is a linear order). But for j-candidate k-approval vectors (approval vectors with exactly k 1's), $k \geq 1$, it is easy to see that the general-case elicitation query complexity is exactly 0 when $j = k$ and is exactly $j - 1$ when $j > k$, but for the single-peaked case, the elicitation query complexity for ik-candidate k-approval vectors is at most $i - 1 + \lceil \log_2 k \rceil$. That is, we get a savings of a multiplicative factor of about k from single-peakedness.

Single-peaked preferences have been studied extensively in political science. We mention just three examples. Ballester and Haeringer [2] provide a precise mathematical characterization of single-peakedness. Lepelley [36] shows that single-peakedness removes some negative results about the relationship between scoring protocols and Condorcet-type criteria. Gailmard et al. [30] discuss Arrow's Theorem on single-peaked domains. We refer the interested reader also to the coverage of single-peaked preferences in the excellent textbook by Austen-Smith and Banks [1].

6. Conclusions and future directions

The central point of this paper is that single-peaked preferences remove many complexity-theoretic shields against control and manipulation. That is, we showed that those shields, already under frequency-of-hardness and approximation attacks from other quarters, for single-peaked preferences didn't even exist in the first place. It follows that when choosing election systems for electorates one suspects will be single-peaked, one must not rely on results for those systems that were proven in the standard, unrestricted preference model.

This paper's work suggests many directions for future efforts. Throughout this paper, single-peaked has meant the unidimensional case. Do the shield removals of this paper hold in, for example, an appropriate two-dimensional (or k-dimensional) analogue? We mention in passing that every profile of n voters voting by linear orders can be embedded into $\mathbb{R}^n$ in such a way that each voter and candidate is a point in $\mathbb{R}^n$ and each voter prefers c_i to c_j if her Euclidean distance to c_i is less than to c_j.

In a human society with a large number of voters, even if one issue, e.g., the economy, is almost completely polarizing the society, there are bound to be a few voters whose preferences are shaped by quite different issues, e.g., a given candidate's religion. So it would be natural to ask whether the shield-evaporation results of this paper can be extended even to societies that are "very nearly" single-peaked (see [22, Section 6] and [14, Section 6] for discussion of nearness to single-peakedness in other contexts). We are currently looking into this issue.

Finally, we mention that recent work of Brandt et al. [12] further explores single-peaked electorates—looking at bribery and at control by partition of voters, and generalizing Theorem 4.4 to each fixed number of candidates.

Acknowledgments

For many helpful comments, discussions, and suggestions, we are grateful to Steven Brams, Felix Brandt, Markus Brill, Felix Fischer, Paul Harrenstein, Jérôme Lang, and Ariel Procaccia, to the anonymous referees of this paper's TARK-09 preliminary version [27], and to *Information and Computation* editor Christos Papadimitriou.

References

[1] D. Austen-Smith, J. Banks, Positive Political Theory II: Strategy and Structure, University of Michigan Press, 2004.
[2] M. Ballester, G. Haeringer, A characterization of the single-peaked domain, Social Choice and Welfare, in press.
[3] S. Barberà, An introduction to strategy-proof social choice functions, Social Choice and Welfare 18 (2001) 619–653.
[4] J. Bartholdi III, J. Orlin, Single transferable vote resists strategic voting, Social Choice and Welfare 8 (1991) 341–354.
[5] J. Bartholdi III, C. Tovey, M. Trick, The computational difficulty of manipulating an election, Social Choice and Welfare 6 (1989) 227–241.
[6] J. Bartholdi III, C. Tovey, M. Trick, How hard is it to control an election? Mathematical and Computer Modeling 16 (1992) 27–40.
[7] J. Bartholdi III, M. Trick, Stable matching with preferences derived from a psychological model, Operations Research Letters 5 (1986) 165–169.
[8] D. Baumeister, G. Erdélyi, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Computational aspects of approval voting, in: J. Laslier, R. Sanver (Eds.), Handbook on Approval Voting, Springer, 2010, pp. 199–251.
[9] D. Black, On the rationale of group decision-making, Journal of Political Economy 56 (1948) 23–34.
[10] D. Black, The Theory of Committees and Elections, Cambridge University Press, 1958.
[11] K. Booth, G. Lueker, Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms, Journal of Computer and System Sciences 13 (1976) 335–379.
[12] F. Brandt, M. Brill, E. Hemaspaandra, L. Hemaspaandra, Bypassing combinatorial protections: Polynomial-time algorithms for single-peaked electorates, in: Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI Press (2010) 715–722.
[13] E. Brelsford, P. Faliszewski, E. Hemaspaandra, H. Schnoor, I. Schnoor, Approximability of manipulating elections, in: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, AAAI Press (2008) 44–49.
[14] V. Conitzer, Eliciting single-peaked preferences using comparison queries, Journal of Artificial Intelligence Research 35 (2009) 161–191.

[15] V. Conitzer, T. Sandholm, Nonexistence of voting rules that are usually hard to manipulate, in: Proceedings of the 21st National Conference on Artificial Intelligence, AAAI Press (2006) 627–634.
[16] V. Conitzer, T. Sandholm, J. Lang,When are elections with few candidates hard to manipulate? Journal of the ACM 54 (2007). Article 14.
[17] O. Davis, M. Hinich, P. Ordeshook, An expository development of a mathematical model of the electoral process, American Political Science Review 54 (1970) 426–448.
[18] J. Doignon, J. Falmagne, A polynomial time algorithm for unidimensional unfolding representations, Journal of Algorithms 16 (1994) 218–233.
[19] E. Ephrati, J. Rosenschein, The Clarke Tax as a consensus mechanism among automated agents, in: Proceedings of the 9th National Conference on Artificial Intelligence, AAAI Press (1991) 173–178.
[20] E. Ephrati, J. Rosenschein, A heuristic technique for multi-agent planning, Annals of Mathematics and Artificial Intelligence 20 (1997) 13–67.
[21] G. Erdélyi, M. Nowak, J. Rothe, Sincere-strategy preference-based approval voting fully resists constructive control and broadly resists destructive control, Mathematical Logic Quarterly 55 (2009) 425–443.
[22] B. Escoffier, J. Lang, M. Öztürk, Single-peaked consistency and its complexity, in: Proceedings of the 18th European Conference on Artificial Intelligence, IOS Press (2008) 366–370.
[23] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, How hard is bribery in elections?, Journal of Artificial Intelligence Research 35 (2009) 485–532.
[24] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Llull and Copeland voting computationally resist bribery and constructive control, Journal of Artificial Intelligence Research 35 (2009) 275–341.
[25] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, A richer understanding of the complexity of election systems, in: S. Ravi, S. Shukla (Eds.), Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz, Springer, 2009, pp. 375–406.
[26] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, The Shield that Never Was: Societies with Single-Peaked Preferences Are More Open to Manipulation and Control, Technical Report arXiv:0909.3257v2 [cs.GT], Computing Research Repository, http://arXiv.org/corr/, 2009, Revised, June 2010.
[27] P. Faliszewski, E. Hemaspaandra, L. Hemaspaandra, J. Rothe, The shield that never was: Societies with single-peaked preferences are more open to manipulation and control, in: Proceedings of the 12th Conference on Theoretical Aspects of Rationality and Knowledge, ACM Press, 2009, pp. 118–127.
[28] E. Friedgut, G. Kalai, N. Nisan, Elections can be manipulated often, in: Proceedings of the 49th IEEE Symposium on Foundations of Computer Science, IEEE Computer Society, 2008, pp. 243–249.
[29] D. Fulkerson, G. Gross, Incidence matrices and interval graphs, Pacific Journal of Mathematics 15 (1965) 835–855.
[30] S. Gailmard, J. Patty, E. Penn, Arrow's theorem on single-peaked domains, in: E. Aragonés, C. Beviá, H. Llavador, N. Schofield (Eds.), The Political Economy of Democracy, Fundación BBVA, 2009, pp. 335–342.
[31] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, 1979.
[32] E. Hemaspaandra, L. Hemaspaandra, Dichotomy for voting systems, Journal of Computer and System Sciences 73 (2007) 73–83.
[33] E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Anyone but him: The complexity of precluding an alternative, Artificial Intelligence 171 (2007) 255–285.
[34] E. Hemaspaandra, L. Hemaspaandra, J. Rothe, Hybrid elections broaden complexity-theoretic resistance to control, Mathematical Logic Quarterly 55 (2009) 397–424.
[35] K. Krehbiel, Pivotal Politics: A Theory of U.S. Lawmaking, University of Chicago Press, 1998.
[36] D. Lepelley, Constant scoring rules, Condorcet criteria, and single-peaked preferences, Economic Theory 7 (1996) 491–500.
[37] R. Niemi, J. Wright, Voting cycles and the structure of individual preferences, Social Choice and Welfare 4 (1987) 173–183.
[38] B. Peleg, P. Sudhölter, Single-peakedness and coalition-proofness, Review of Economic Design 4 (1999) 381–387.
[39] D. Pennock, E. Horvitz, 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, AAAI Press (2000) 729–734.
[40] K. Poole, H. Rosenthal, Congress: A Political-Economic History of Roll-Call Voting, Oxford University Press, 1997.
[41] A. Procaccia, J. Rosenschein, Junta distributions and the average-case complexity of manipulating elections, Journal of Artificial Intelligence Research 28 (2007) 157–181.
[42] M. Trick, Recognizing single-peaked preferences on a tree, Mathematical Social Sciences 17 (1989) 329–334.
[43] T. Walsh, Uncertainty in preference elicitation and aggregation, in: Proceedings of the 22nd AAAI Conference on Artificial Intelligence, AAAI Press (2007) 3–8.
[44] L. Xia, V. Conitzer, Generalized scoring rules and the frequency of coalitional manipulability, in: Proceedings of the 9th ACM Conference on Electronic Commerce, ACM Press (2008) 109–118.
[45] L. Xia, V. Conitzer, A sufficient condition for voting rules to be frequently manipulable, in: Proceedings of the 9th ACM Conference on Electronic Commerce, ACM Press (2008) 99–108.