review articles

Computational complexity may truly be the shield against election manipulation.

BY PIOTR FALISZEWSKI, EDITH HEMASPAANDRA, AND LANE A. HEMASPAANDRA

Using Complexity to Protect Elections

FOR THOUSANDS OF years, people—and more recently, electronic agents—have been conducting elections. And surely for just as long, people—or more recently, electronic agents—have been trying to affect the outcomes of those elections. Such attempts take many forms. Often and naturally, actors may seek to change the structure of the election, for example, by attracting new voters, suppressing turnout, recruiting candidates, or setting election district boundaries. Sometimes voters may even be bribed to vote a certain way. And a voter may try to manipulate an election by casting an insincere vote that may yield a more favorable outcome than would the voter's sincere vote: Not all people who preferred Ralph Nader in the 2004 U.S. presidential election actually voted for him.

One might hope that by choosing a particularly wonderful election system, one can perfectly block

such attacks. However, classic work from economics and political science proves that every reasonable election system sometimes gives voters an incentive to vote insincerely (see Duggan[17] and the references therein). Reasonable election systems cannot make manipulation impossible. However, they can make manipulation computationally infeasible.

This article is a nontechnical introduction to a startling approach to protecting elections: using computational complexity as a shield. This approach seeks to make the task of whoever is trying to affect the election *computationally* prohibitive. To better understand the cases in which such protection cannot be achieved, researchers in this area also spend much of their time working for the Dark Side: trying to build polynomial-time algorithms to attack election systems.

This complexity-based approach to protecting elections was pioneered in a stunning set of papers, about two decades ago, by Bartholdi, Orlin, Tovey, and Trick.[2,3,5] The intellectual fire they lit smoldered for quite a while, but in recent years has burst into open flame. Computational complexity may truly be the key to defending elections from manipulation.

Preliminaries and the Complexity of the Winner Problem

In the introduction, we focused on

> » **key insights**

- ■ **Algorithms can be used to seek attacks on elections, and complexity can serve to protect elections from attacks. For some election systems, manipulation has been proven NP-hard.**

- ■ **Dichotomy theorems pinpoint what it is about an election system that makes it computationally resistant to manipulation.**

- ■ **It is natural to consider an election system's computational weaknesses and strengths as one factor, among many, when selecting a system for a given task. In particular, one must consider which types of attacks one most needs to thwart.**
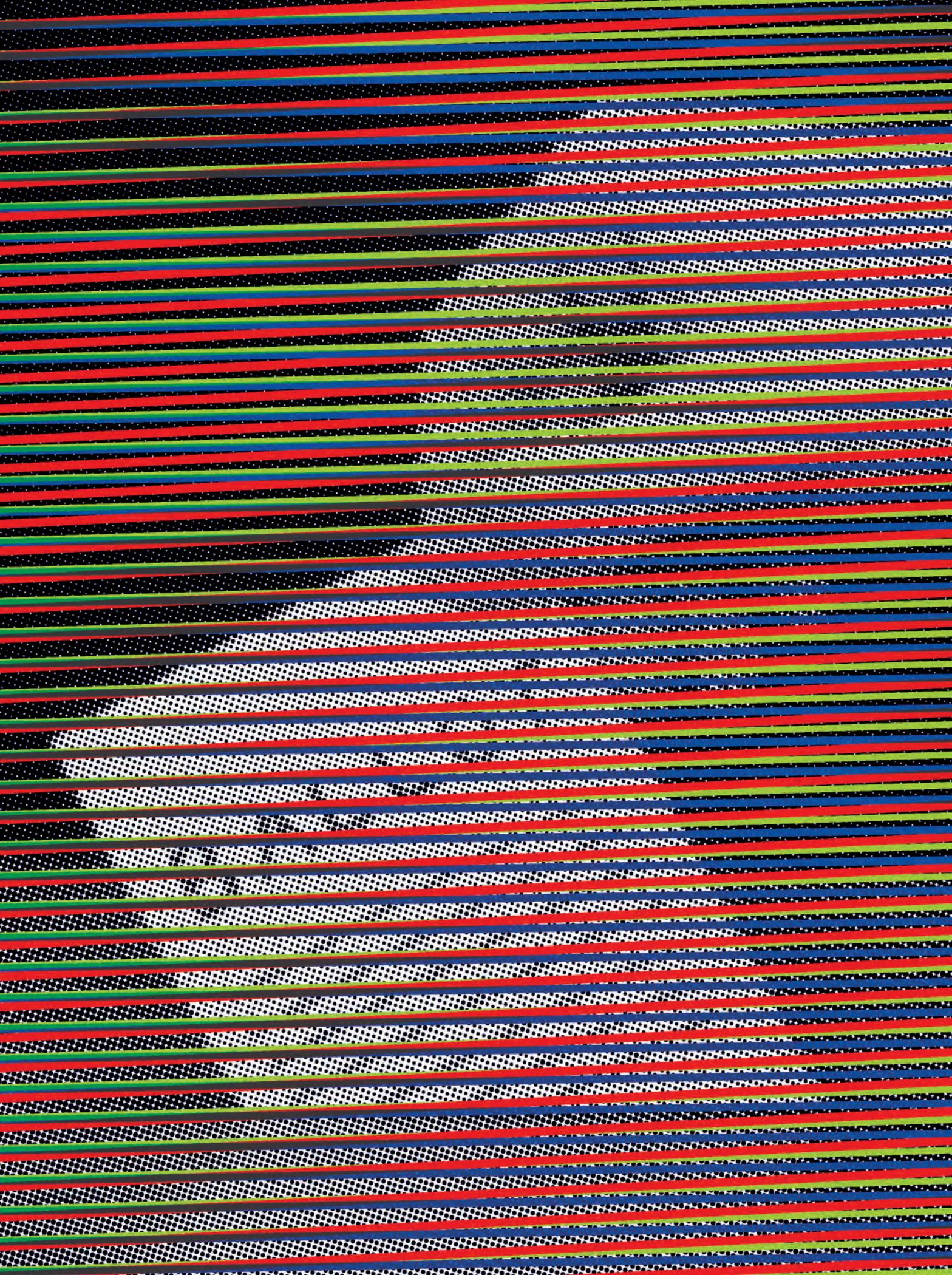
protecting elections, rather than on why and in what settings elections are used for aggregating preferences in the first place. The latter issue could itself fill a survey—but not this survey. However, before moving on we briefly mention a few varied examples of how elections can be useful in aggregating preference. In daily life, humans use elections to aggregate preferences in tasks ranging from citizens choosing their political representatives to an academic department's faculty members selecting which job candidate to hire to conference business meeting attendees selecting future locations for their conference. In electronic settings, elections often can take on quite different, yet also interesting and important, challenges. For example, one can build a metasearch engine based on combining underlying search engines, in order to seek better results and be more resistant to "Web spam."[18] One can use voting as an approach to building recommender systems[41] and to planning.[20] Voting was already very important before computers and the internet existed, and in the modern world, where multiagent settings abound, the importance of voting is greater still.

In this article, we will discuss the successes and failures to date in using complexity to defend against three important classes of attacks on election systems: (structural) control attacks, (voter) manipulation, and bribery. In these three settings, high computational complexity is the goal. But first, we briefly discuss a case so surprising that one might not even think of it, namely, the case in which an election system is so complex that even determining who won is intractable.

We must first introduce the model of elections we will use throughout this article. While doing so, we will also define some election systems, such as plurality rule. An election consists of a candidate set C and a list V of votes (ballots) over those candidates. In almost all the election systems we discuss, a vote is simply a strict ordering of all the candidates, for example, "Nader > Gore > Bush" if the voter likes Nader most, Gore next most, and Bush least. An exception is approval voting, in which each vote is a bit-vector giving a thumbs-up or thumbs-down to each candidate.

An election system is simply a map-

Voting was already very important before computers and the Internet existed, and in the modern world, where multiagent settings abound, the importance of voting is greater still.

ping from (C, V) to a "winner set" W, $\emptyset \subseteq W \subseteq C$. Perhaps the most famous and common election system is plurality, in which each candidate who most often comes at the top of voters' orders is put into W. We will focus quite a bit on plurality in this article, since it has been extensively studied with respect to using complexity to protect elections. Plurality is itself a special case of a broad class of election systems known as scoring systems or scoring-rule systems. In these, each candidate gets from each voter a certain number of points based on where he or she falls in the voter's ordering, and whoever gets the most points wins. For example, the scoring point system for plurality (in k-candidate elections) is that a voter's favorite candidate gets one point from that voter and the other $k–1$ candidates get zero points from that voter. In the Borda election system, proposed in the 18th century, the points from favorite to least favorite are $k–1, k–2, . . . , 0$. In veto elections, the points are $1, 1, 1, . . . , 1, 0$; that is, the voter in effect votes against one candidate. Scoring systems are a flexible, important class of voting systems and, as we will see, they are a class whose manipulation complexity (for fixed numbers of candidates) is completely analyzed. There are many other important election systems, but to move the article along, we will introduce them as we need.

An election system that immediately merits note is the Condorcet rule. In Condorcet elections, a candidate is a winner exactly if he or she beats each other candidate in head-to-head majority-rule elections under the voters' preferences. Consider the election shown in Figure 1. In that election there is no Condorcet winner, since 🧑 is beaten by 🧑 3-to-1, 🧑 is beaten by 🧑 4-to-0, and

Figure 1. An election.

in a 2-to-2 tie, 🐢 fails to beat 🐸.

Although this example shows that Condorcet elections sometimes have no winner, some election systems—the so-called Condorcet-consistent systems—so value the naturalness of the notion of being a Condorcet winner that they ensure that when a Condorcet winner exists, he or she is the winner in their system. One particularly interesting system is the election system proposed by the famous author and mathematician Lewis Carroll in 1876. Carroll took an approach that should warm the hearts of computer scientists. He said, in effect, that whoever had the smallest edit distance from being a Condorcet winner was the winner in his election system. His edit distance was with respect to the number of sequential exchanges of adjacent candidates in voter orderings. So in the Figure 1 example, 🐢 and 🐸 tie as Carroll winners, since either of them with one adjacent exchange can become a Condorcet winner (for example, if we by one exchange turn voter 1's preference list into 🐸 > 🐢 > 🐱 > 🐹, then 🐸 becomes a Condorcet winner), but 🐱 for example would take seven adjacent exchanges to become a Condorcet winner.

Lewis Carroll's system is quite lovely in that it focuses on the closeness to Condorcet winnerhood. Carroll's paper has been included in books collecting the most important social choice papers of all time. However, Carroll's sys-

tem has one glaring flaw: It is computationally intractable to tell who won! This was first shown in a paper by Bartholdi, Tovey, and Trick,[4] who showed that this problem was **NP**-hard. Later, Hemaspaandra, Hemaspaandra, and Rothe[30] precisely classified the problem's complexity as "complete" (that is, in a certain formal sense the hardest problem) for the class of problems that can be solved by parallel access to **NP** (a class that forms the Θ_2^p level of the polynomial hierarchy).[a]

On its face, this result is a disaster for Lewis Carroll's election system. Although we want manipulation of elections to be difficult, we do not want to achieve this by migrating to election systems so opaque that we cannot efficiently compute who won.

This disaster may not be quite as severe as it first seems. Recent work on Lewis Carroll elections seeks to slip around the edges of the just-mentioned intractability result. In particular, two recent papers show that simple

---

a We will not provide here a discussion of *NP*-hardness/*NP*-completeness/Θ_2^p-completeness, but suffice it to say that complexity theorists broadly believe any problem that has any one of these properties is intractable, that is, does not have a deterministic polynomial-time algorithm. However, these notions are worst-case notions. In the section "Using Complexity to Block Election Manipulation" we will discuss how their worst-case nature is itself a worry when using them to protect election systems.

polynomial-time greedy algorithms correctly find the Lewis Carroll winner all but an asymptotically exponentially vanishing portion of the time when the number of voters is more than quadratically larger than the number of candidates and the inputs are drawn from the uniform probability distribution.[35,39] In fact, that algorithm can even be made "self-knowingly correct"—it almost always declares that its answer is correct, and when it does so it is never wrong.[35] Another way of arguably bypassing the hardness results for the Lewis Carroll winner problem is through approximation algorithms. For example, Caragiannis et al.[9] have recently developed two approximation algorithms for computing candidates' scores in Carroll's system. And a third way to sidestep the hardness results is to change the framework, namely, to assume that the number of candidates or the number of voters is bounded by a fixed constant, and to seek polynomial-time algorithms in that setting.[b] The seminal paper of Bartholdi, Tovey, and Trick[4] successfully pursued this line, as

---

b Many real-life settings have relatively few candidates. And a particularly interesting setting with few voters but many candidates comes from Dwork et al.,[18] who suggested building a search engine for the Web that would simply query other search engines and then conduct an election given the search engines' answers as votes.

Table 1. The computational complexity of control in Condorcet, Copeland, Llull, and plurality elections.

The results regarding constructive control in Condorcet and plurality elections are due to Bartholdi et al.,[5] the results on destructive control for Condorcet and plurality are due to Hemaspaandra et al.,[31] and the results regarding Llull and Copeland are due to Faliszewski et al.[25] Adding (Unlimited Number of) Candidates has not been explicitly studied in Bartholdi et al.[5] and Hemaspaandra et al.,[31] but the results on this for Condorcet and plurality elections are corollaries to these papers' proofs.

Election System / Control Type	Condorcet		Copeland		Llull		Plurality	
	Const. Control	Dest. Control	Const. Control	Dest. Control	Const. Control	Dest. Control	Const. Control	Dest. Control
Adding (Unlimited Number of) Candidates	I	V	R	V	V	V	R	R
Adding Candidates	I	V	R	V	R	V	R	R
Deleting Candidates	V	I	R	V	R	V	R	R
Run-off Partition of Candidates (Ties Promote)	V	I	R	V	R	V	R	R
Run-off Partition of Candidates (Ties Eliminate)	V	I	R	V	R	V	R	R
Partition of Candidates (Ties Promote)	V	I	R	V	R	V	R	R
Partition of Candidates (Ties Eliminate)	V	I	R	V	R	V	R	R
Partition of Voters (Ties Eliminate)	R	V	R	R	R	R	V	V
Partition of Voters (Ties Promote)	R	V	R	R	R	R	R	R
Adding Voters	R	V	R	R	R	R	V	V
Deleting Voters	R	V	R	R	R	R	V	V

Figure 2. Ramon Llull, 13th-century mystic and philosopher.

have more recent papers.[7,11,24, 25] However-er, their polynomial-time algorithms sometimes involve truly astronomical multiplicative constants.

Finally, we mention that in the years since the work showing Lewis Carroll's election system to have a winner problem that is complete for parallel access to **NP**, a number of other systems, most notably those of Kemeny and Young, have also been shown to be complete for parallel access to **NP**.[33,43] Naturally, researchers have sought to bypass these hardness results as well (for examples, see[6,9,12,36]).

Using Complexity to Block Election Control

Can our choice of election systems—and not merely nasty ones with hard winner problems but rather natural ones with polynomial-time winner problems—be used to make influencing election outcomes costly? We start the discussion of this issue by considering problems of election *control*, introduced by Bartholdi, Tovey, and Trick[5] in 1992. In election control, some actor who has complete knowledge of all the votes seeks to achieve a desired outcome—either making a favored candidate be the sole winner ("constructive control") or precluding a despised candidate from being a unique winner ("destructive control")—via changing the structure of the election. The types of structural changes that Bartholdi, Tovey, and Trick proposed are adding or deleting candidates, adding or deleting voters, or partitioning candidates or voters into a two-round election structure.

Between these types and the different tie-breaking rules that can be used to decide which candidates move forward from the preliminary rounds of two-round elections in the case of ties (that is, whether all the tied people move forward or none of them do), there now are eleven types of control that are typically studied—each having both a constructive and a destructive version.

For reasons of space we will not define all 11 types here. We will define one type explicitly and will mention the motivations behind most of the others. Let us consider Control by Deleting Voters. In this control scenario, the input to the problem is the election (C, V), a candidate $p \in C$, and an integer k. The question is whether by removing at most k votes from V one can make p be the sole winner (for the constructive case) or can preclude p from being a unique winner (for the destructive case). Control by Deleting Voters is loosely inspired by vote suppression: It is asking whether by the targeted suppression of at most k votes the given goal can be reached. (By discussing vote suppression we are in no way endorsing it, and indeed we are discussing paths toward making it computationally infeasible.) So, for a given election system E, we are interested in the complexity of the set composed of all inputs (C, V, p, k) for which the goal can be reached.[c]

---

c As to *who* is seeking to do the control, that is external to the model. For example, it can be some central authority or a candidate's campaign committee. In fact, in the real world there often are competing control actors. But results we will soon cover show that even a single control actor faces a computationally infeasible problem. Also, the reader may naturally feel uncomfortable with the model's assumption that the

The other control types similarly are motivated as abstractions of real-world actions—many far more savory than vote suppression. For example, Control by Adding Voters abstracts such actions as get-out-the-vote drives, positive advertising campaigns, providing vans to drive elderly people to the polls, registration drives, and so on. Control by Adding Candidates and Control by Deleting Candidates reflect the effect of recruiting candidates into—and pressuring them to withdraw from—the race. The memory of the 2000 U.S. presidential race suggests that whether a given small-party candidate—say Ralph Nader—enters a race can change the outcome. The partition models loosely capture other behaviors, such as gerrymandering.

Table 1 summarized the constructive and destructive control results for four election systems whose behavior is completely known: Plurality, Condorcet, Llull, and Copeland. The mystic and philosopher Ramon Llull (Figure 2) defined the Llull system in the 1200s, and the Copeland system is a closely related system defined in modern times. In both of these systems one considers each pair of candidates and awards one point to the winner in their head-to-head majority-rule contest, and if the head-to-head contest is a tie, in Copeland each gets half a point but in Llull each still gets one point. So, for example, in Copeland one gets $\|C\|-1$ points exactly if one is a Condorcet winner. The Llull/Copeland system is used in the group stage

---

control actor knows all the votes of all the voters. But note that that just makes the "shield" results stronger: they show that even if one had perfect information about the votes, finding a control action is *still* intractable.

Figure 3. The points assigned by the Llull/Copeland systems in the head-to-head contests of the election of Figure 1.
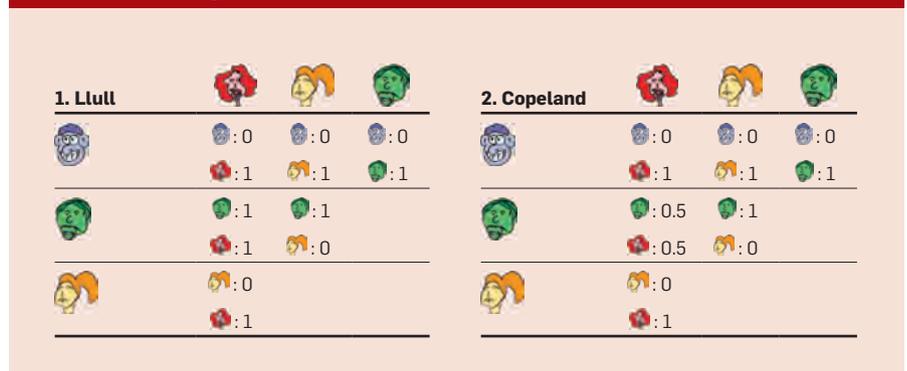
of the World Cup soccer tournament, except there (after rescaling) wins get one point and ties get one third of a point. Figure 3 shows how the election from Figure 1 comes out under the Llull and Copeland systems. In Table 1, I (immunity) means one can never change the outcome with that type of control attack—a dream case; R (resistance) means it is **NP**-hard to determine whether a given instance can be successfully attacked—still a quite good case; and V (vulnerability) means there is a polynomial-time algorithm to detect whether there is a successful attack of the given type (and indeed to produce the exact attack)—the case we wish to avoid.

Remarkably, given that Llull created his system in the 1200s, among all natural systems based on preference orders, Llull and Copeland are the systems that currently have the greatest numbers of proven resistances to control. As one can see from Table 1, Copeland is perfectly resistant to the constructive control types and to all voter-related control types (but is vulnerable to the destructive, candidate-related control types). And Llull's 13th-century system is almost as good. Ramon Llull, the mystic, truly was ahead of his time.

If one wants an even greater number of resistances than Copeland/Llull provides, one currently can do that in two different ways. Recently, Erdélyi, Nowak, and Rothe[22] showed that a voting system whose votes are in a different, richer model—each voter provides both an approval vector and a strict ordering—has a greater number of control resistances, although in achieving that it loses some of the particular resistances of Copeland/Llull. And Hemaspaandra, Hemaspaandra, and Rothe[32] constructed a hybridization scheme that allows one to build an election system whose winner problem—like the winner problem of all four systems from Table 1—is computationally easy, yet the system is resistant to all 22 control attacks. Unfortunately, that election system is in a somewhat tricky manner "built on top of" other systems each of which will in some cases determine the winner, and so the system lacks the attractiveness and transparency that real-world voters reasonably expect.

The current push-pull between using complexity as a shield and seeking holes in and paths around that shield is a natural part of the drama of science.
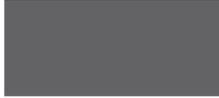
To conclude our discussion of control, we mention one other setting, that of choosing a whole assembly or committee through an election. Such assembly-election settings introduce a range of new challenges. For example, the voters will have preferences over assemblies rather than over individual candidates. We point the reader to the work of Meir et al.[40] for results on the complexity of controlling elections of this type.

Using Complexity to Block Election Manipulation

Manipulation is often used informally as a broad term for attempts to affect election outcomes. But in the literature, manipulation is also used to refer just to the particular attack in which a voter or a coalition of voters seeks to cast their votes in such a way as to obtain a desired outcome, for example, making some candidate win. In formulating such problems, one often studies the case in which each voter has a weight, as is the case in the electoral college and in stockholder votes. The input to such problems consists of the weights of all voters, the votes of the nonmanipulators, and the candidate the manipulators are trying to make a winner.

Manipulation problems have been studied more extensively than either control or bribery problems, and so the literature is too broad to survey in any detail. But we now briefly mention a few of the key themes in this study, including using complexity to protect, using algorithms to attack, studying approximations to bypass protections, and analyzing manipulation properties of random elections.

The seminal papers on complexity of manipulation are those of Bartholdi, Orlin, Tovey, and Trick.[2,3] Bartholdi, Tovey, and Trick[3] gave polynomial-time algorithms for manipulation and proved a hardness-of-manipulation result (regarding so-called second-order Copeland voting). Bartholdi and Orlin[2] showed that for "single transferable vote," a system that is used for some countries' elections, whether a given voter can manipulate the election is **NP**-complete, even in the unweighted case.

Even if election systems are proven intractable to manipulate in general, it remains possible that if one allows only

a certain number of candidates, the manipulation problem becomes easy. Conitzer, Sandholm, and Lang[15] provide a detailed study of this behavior, showing for each of many election systems the exact number of candidates necessary to make its (constructive, weighted, coalitional) manipulation problem computationally infeasible. For example, in this setting manipulation is easy for Borda with up to two candidates, but becomes infeasible when restricted even to three candidates.

In contrast, it is well known that manipulation is simple for plurality elections regardless of the number of candidates. That is unfortunate, since plurality elections are the most common and most important elections in the real world.

What holds for scoring-rule election systems other than plurality? One could try analyzing scoring systems one at a time to see which are subject to manipulation, but it might be a long slog since there are an infinite number of scoring systems. This motivates us to look toward an excellent general goal: finding a dichotomy theorem that in one fell swoop pinpoints what it is about an election system that makes it vulnerable to manipulation or that makes manipulation computationally prohibitive. For scoring systems, this was achieved in Hemaspaandra and Hemaspaandra[29] (see also the closely related work[15,42]),

Figure 4. An example of a weighted plurality election.

Each bar represents a weighted vote for a particular candidate. We can make p a winner by bribing the weight-5 voter to vote for p, but bribing only the heaviest voter to vote for p would not be sufficient.
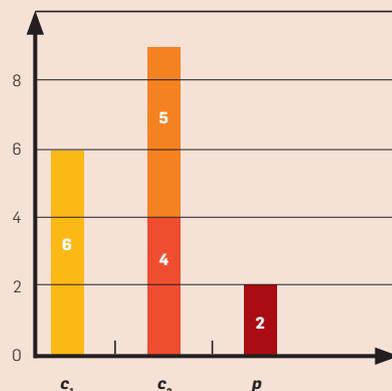
which showed that scoring systems are **NP**-complete to manipulate (in the weighted setting) precisely if they allow "diversity of dislike" (that is, the point values for the second favorite and least favorite candidates differ), and that all other scoring systems are easy to manipulate. From this it follows that the only easily manipulable scoring systems are an infinite collection of trivial systems, plurality, and an infinite collection of systems that are disguised, transformed versions of plurality; all other scoring systems are **NP**-hard to manipulate.

There has been an intense effort to circumvent such hardness results. Indeed, the seminal paper on manipulation[3] provided a greedy single-voter manipulation algorithm that was later proved to also work in an interesting range of coalitional-manipulation settings.[42,49] An influential paper of Conitzer and Sandholm[14] shows that voting systems and distributions that on a large probability weight of the inputs satisfy certain conditions have a manipulability-detection algorithm that is correct on at least that same set of inputs. A different line of research focuses on analyzing the probability with which a randomly selected election is susceptible to a given form of manipulation.[16,28,47,48] In the standard probabilistic model used in this line of work,[d] for many natural election systems the probability that a voter can affect the result of an election by simply casting a random vote is small but nonnegligible.

This work is motivated by perhaps the greatest single worry related to using **NP**-hardness to protect elections— a worry that applies to **NP**-hardness results not just about manipulation, but also about control and bribery. That worry is that **NP**-hardness is a worst-case theory, and it is in concept possible that **NP**-hard sets may be easily solved on many of their input instances even if **P** and **NP** differ.[e] Levin has

famously developed the theory of average-case **NP**-hardness,[37] and although that theory is difficult to apply and is tied to what distributions one uses, it would be extremely interesting to establish that the manipulation, control, and bribery problems for important election systems are average-case **NP**-hard with respect to some appropriate and compellingly natural distribution.

A very exciting new path toward circumventing hardness-of-manipulation results (and, potentially, toward more generally circumventing hardness results about election-related issues) is to look at restricted domains for the collections of votes the electorate may cast. In particular, there is a very important political science notion called "single-peaked preferences," in which the candidates are modeled along an axis, such as liberal to conservative, and as one goes away from each voter's most preferred candidate in either of the axis's directions the voter prefers the candidates less and less. Walsh[46] raised the fascinating question of whether hard election-manipulation problems remain hard even for electorates that follow the single-peaked model, and he provided natural examples in which manipulation problems remain hard even when restricted to single-peaked electorates. In contrast, and inspired by a different part of Walsh's paper that showed some profile completion problems are easy for single-peaked electorates, a recent paper by Faliszewski et al.[27] shows that for single-peaked electorates many **NP**-hard manipulation and control problems have polynomial-time algorithms. The point of—and threat of—this research line is that for electorates that are single-peaked,

---

d   This model is called *impartial culture*. In impartial culture each vote is chosen uniformly at random from the set of all permutations of the candidates.

e   There are a number of results in theoretical computer science that are related to this issue, while as a practical matter not resolving it for the concrete cases we care about. For example, by an easy "padding" trick one can see that *every* **NP**-hard set can have its instances transformed into questions about (in the jargon,

---

can be many-one polynomial-time reduced to) a set that is easy on overwhelmingly many of its instances.[21] Unfortunately, this does not necessarily imply that the original set is easy on overwhelmingly many of its instances. In fact, it is known that relative to a random "oracle" (black box), there are **NP** sets on which no polynomial-time heuristic algorithm can do well.[34] Also, it is well known that if any **NP**-hard set has a polynomial-time heuristic algorithm that is correct on all but a "sparse" amount of its input, then **P** = **NP**.[44] However, "sparse" in that research line is so small as to not reassure us here. And, finally, there has been much interest in distributions, problems, and settings that remove the gap between worst-case and average-case complexities.[1,38]

NP-hardness results simply may fail to hold. And the reason that can happen is the assumption of single-peaked preferences is so restrictive that it can rule out some of the collections of votes used in the outputs of reductions in general-case **NP**-hardness proofs.

Yet another path toward circumventing hardness-of-manipulation results leads to relaxing the notion of solving a manipulation problem. Procaccia and Rosenschein[42] initiated this approach by showing that the heuristic from the seminal work of Bartholdi, Tovey, and Trick,[3] when extended to a coalitional manipulation setting, works correctly on an interesting class of scoring-system manipulation instances. By an even more careful analysis, together with Zuckerman, they later extended this result to a number of other election systems,[49] and they obtained approximation results and results that for manipulable instances are guaranteed to return a manipulation that will work if one is allowed to add a certain number and weight of additional manipulators. Brelsford et al.[8] provide their own framework for studying approximability of manipulation problems (as well as approximability of bribery and control problems) and for a large class of scoring systems gives approximation algorithms for manipulation.

Returning to playing defense, what can we do if a system has a polynomial-time manipulation algorithm? Can we somehow feed the system a can of spinach and turn it fearsome? To a surprising extent the answer is yes, as studied in work of Conitzer and Sandholm[13] and Elkind and Lipmaa.[19] They variously do this by adding an elimination "pre-round" (that may or may not be based on a hypothetical one-way function) or by changing the election into a long series of rounds of candidate elimination. The good news is that this approach often boosts the complexity, and the bad news is that these multiround election systems are simply not the same intuitively attractive animals that they are built from.

Using Complexity to Block Bribery in Elections

The complexity-theoretic study of bribery in elections was proposed by Faliszewski, Hemaspaandra, and He-

maspaandra,[24] and started far more recently than did the complexity-theoretic study of control and manipulation of elections. Bribery comes in many variants, but the basic pattern is just what the term brings to mind. The briber has a certain budget, the voters (who depending on the model may or may not have weights) each have a price for which their vote can be bought, and depending on the model voters may or may not be required to each have unit cost (the former case is referred to as the "without prices" case). And the question is whether the briber can achieve his or her goal—typically, to make a preferred candidate p be a winner—within the budget. Note that bribery has aspects of both control and manipulation. Like some types of control one has to choose which collection of voters to act on, but like manipulation one is altering votes.

For reasons of space, we cover bribery only briefly. We do so by giving a few examples focusing on plurality elections and Llull elections.

For plurality elections, the complexity of bribery turns out to be very sensitive to the model. For plurality, bribery is **NP**-complete when voters have weights and prices, but is in polynomial time if voters have only weights, only prices, or neither weights nor prices.[24,f] For the weighted and the weighted-and-priced cases, these results can be extended to dichotomy theorems that completely classify which scoring-rule election systems have **NP**-complete bribery problems and which have feasible bribery problems.[24] Also, for plurality, there is an efficient algorithm that can approximately solve the problem up to any given precision[23]—a so-called fully polynomial-time approximation scheme.

For Llull elections, the results again are very sensitive to the model. On one hand, both with and without weights, and both with and without voter prices, the bribery problem for Llull elections is **NP**-complete. On the other

hand, if one changes one's model and associates a cost not to each voter, but rather to each pairwise preference of each voter (so the more one changes a given voter's vote, the more one has to pay—so-called "microbribery"), Llull bribery (without weights) can be done, in a slightly different model that allows irrational preferences, in polynomial time.[25]

Summary

In this article, we discussed some of the main streams—control, manipulation, and bribery—in the study of how complexity can be used as a shield to protect elections (see Faliszewski et al.[26] for a more technical survey). This line was started by the striking insight of Bartholdi, Orlin, Tovey, and Trick (see also Simon[45] for even earlier roots) that although economics proves we cannot make manipulation impossible, we can seek to make it computationally infeasible. As we have seen, many hardness results have been obtained, as have many polynomial-time attacks. Election systems and settings vary greatly in the behaviors one can establish. It is natural to consider an election system's computational weaknesses and strengths, as one factor among many, when choosing an election system for a given task, and in particular to choose a system carefully in light of the types of attacks one most needs it to thwart. Yet the work on computational protection of elections has also energized the search for end runs around that protection, such as approximation algorithms and heuristics having provably frequent good performance, and one must also worry about such potential end runs when making one's election-system choice.

This work all falls within the emerging area known as computational social choice (see Chevaleyre et al.[10] for a superb survey), an area that links AI, systems, and theory within computer science, as well as economics, political science, mathematics, and operations research. Elections have been important for thousands of years, and with the current and anticipated increase of electronic agency, elections become more important—and more open to attacks—with each passing year. The current push-pull between using complexity

---

f   The bribery algorithms are far from trivial. For example, Figure 4 shows an election (without prices) where the very natural heuristic of first bribing the heaviest voter yields a suboptimal solution. Similarly, it is easy to find examples where bribing the heaviest voter of a current winner does not lead to an optimal solution.

as a shield and seeking holes in and paths around that shield is a natural, exciting part of the drama of science, and is likely to continue for decades to come as new models, techniques, and attacks are formulated and studied. This study will clearly benefit from the broadest possible participation, and we urge any interested readers—and most especially those early in their careers—to bring their own time and skills to bear on the many problems that glimmer in the young, important, challenging study of the complexity of elections.

Acknowledgments

References

1. Ajtai, M. Worst-case complexity, average-case complexity and lattice problems. *Documenta Mathematica*, Extra Volume ICM III (1998), 421–428.
2. Bartholdi, III, J. and Orlin, J. Single transferable vote resists strategic voting. *Social Choice and Welfare 8*, 4 (1991) 341–354.
3. Bartholdi, III, J., Tovey, C. and Trick, M. The computational difficulty of manipulating an election. *Social Choice and Welfare 6*, 3 (1989) 227–241.
4. Bartholdi, III, J., Tovey, C. and Trick, M. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare 6*, 2 (1989), 157–165.
5. Bartholdi, III, J., Tovey, C. and Trick, M. How hard is it to control an election? *Mathematical and Computer Modeling 16*, 8/9 (1992), 27–40.
6. Betzler, N., Fellows, M., Guo, J., Niedermeier, R. and Rosamond, F. Fixed-parameter algorithms for Kemeny scores. In *Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management*. Lecture Notes in Computer Science #5034 (June 2008). Springer-Verlag, 60–71.
7. Betzler, N., Guo, J., Niedermeier, R. Parameterized computational complexity of Dodgson and Young elections. In *Proceedings of the 11th Scandinavian Workshop on Algorithm Theory*. Lecture Notes in Computer Science #5124 (July 2008). Springer-Verlag, 402–413.
8. Brelsford, E., Faliszewski, P., Hemaspaandra, E., Schnoor, H., and Schnoor, I. Approximability of manipulating elections. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence* (July 2008). AAAI Press, 44–49.
9. Caragiannis, I., Covey, J., Feldman, M., Homan, C. Kaklamanis, C., Karanikolas, N., Procaccia, A. and Rosenschein, J. On the approximability of Dodgson and Young elections. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms* (Jan. 2009). Society for Industrial and Applied Mathematics, 1058–1067.
10. Chevaleyre, Y., Endriss, U., Lang, J. and Maudet, N. A short introduction to computational social choice. In *Proceedings of the 33rd International Conference on Current Trends in Theory and Practice of Computer Science*. Lecture Notes in Computer Science #4362 (Jan. 2007). Springer-Verlag, 51–69.
11. Christian, R., Fellows, M., Rosamond, F. and Slinko, A. On complexity of lobbying in multiple referenda. *Review of Economic Design 11*, 3 (2007), 217–224.
12. Conitzer, V., Davenport, A. and Kalagnanam, J. Improved bounds for computing Kemeny rankings. In *Proceedings of the 21st National Conference on Artificial Intelligence* (July 2006). AAAI Press, 620–626.
13. Conitzer, V. and Sandholm, T. Universal voting protocol tweaks to make manipulation hard. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (Aug. 2003). Morgan Kaufmann, 781–788.
14. Conitzer, V. and Sandholm, T. Nonexistence of voting rules that are usually hard to manipulate. In *Proceedings of the 21st National Conference on Artificial Intelligence* (July 2006). AAAI Press, 627–634.
15. Conitzer, V., Sandholm, T. and Lang, J. When are elections with few candidates hard to manipulate? *Journal of the ACM 54*, 3 (2007), Article 14.
16. Dobzinski, S. and Procaccia, A. Frequent manipulability of elections: The case of two voters. In *Proceedings of the 4th International Workshop on Internet and Network Economics*. (Dec. 2008). Springer-Verlag *Lecture Notes in Computer Science #5385*, 653–664.
17. Duggan, J. and Schwartz, T. Strategic manipulability without resoluteness or shared beliefs: Gibbard–Satterthwaite generalized. *Social Choice and Welfare 17*, 1 (2000), 85–93
18. Dwork, C., Kumar, R., Naor, M. and Sivakumar, D. Rank aggregation methods for the Web. In *Proceedings of the 10th International World Wide Web Conference* (Mar. 2001). ACM Press, NY, 613–622.
19. Elkind, E. and Lipmaa, H. Hybrid voting protocols and hardness of manipulation. In *Proceedings of the 16th Annual International Symposium on Algorithms and Computation* (Dec. 2005). Springer-Verlag *Lecture Notes in Computer Science #3872*, 206–215.
20. Ephrati, E. and Rosenschein, J. A heuristic technique for multi-agent planning. *Annals of Mathematics and Artificial Intelligence 20*, 1-4 (1997), 13–67.
21. Erdélyi, G., Hemaspaandra, L., Rothe, J. and Spakowski, H. Generalized juntas and NP-hard sets. *Theoretical Computer Science 410*, 38–40 (2009), 3995–4000.
22. Erdélyi, G., Nowak, M. and Rothe, J. Sincere-strategy preference-based approval voting fully resists constructive control and broadly resists destructive control. *Mathematical Logic Quarterly 55*, 4 (2009), 425–443.
23. Faliszewski, P. Nonuniform bribery (short paper). In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems* (May 2008), 1569–1572.
24. Faliszewski, P., Hemaspaandra, E. and Hemaspaandra, L. How hard is bribery in elections? *Journal of Artificial Intelligence Research 35* (2009), 485–532.
25. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L. and Rothe, J. Llull and Copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research 35* (2009), 275–341.
26. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L. and Rothe, J. A richer understanding of the complexity of election systems. In *Fundamental Problems in Computing: Essays in Honor of Professor Daniel J. Rosenkrantz*. S. Ravi and S. Shukla, eds. Springer, 2009, 375–406.
27. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L. and Rothe, J. 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* (July 2009). ACM Digital Library, 118–127.
28. Friedgut, E., Kalai, G. and Nisan, N. Elections can be manipulated often. In *Proceedings of the 49th IEEE Symposium on Foundations of Computer Science* (Oct. 2008). IEEE Computer Society, 243–249.
29. Hemaspaandra, E., Hemaspaandra, L. Dichotomy for voting systems. *Journal of Computer and System Sciences 73*, 1 (2007), 73–83.
30. Hemaspaandra, E., Hemaspaandra, L. and Rothe, J. Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP. *Journal of the ACM 44*, 6 (1997), 806–825.
31. Hemaspaandra, E., Hemaspaandra, L. and Rothe, J. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence 171*, 5–6 (2007), 255–285.
32. Hemaspaandra, E., Hemaspaandra, L. and Rothe, J. Hybrid elections broaden complexity-theoretic resistance to control. *Mathematical Logic Quarterly 55*, 4 (2009), 397–424.
33. Hemaspaandra, E., Spakowski, H. and Vogel, J. The complexity of Kemeny elections. *Theoretical Computer Science 349*, 3 (2005), 382–391.
34. Hemaspaandra, L. and Zimand, M. Strong self-reducibility precludes strong immunity. *Mathematical Systems Theory 29*, 5 (1996), 535–548.
35. Homan, C. and Hemaspaandra, L. Guarantees for the success frequency of an algorithm for finding Dodgson-election winners. *Journal of Heuristics 15*, 4 (2009), 403–423.
36. Kenyon-Mathieu, C. and Schudy, W. How to rank with few errors. In *Proceedings of the 39th ACM Symposium on Theory of Computing* (June 2007). ACM Press, 95–103.
37. Levin, L. Average case complete problems. *SIAM Journal on Computing 15*, 1 (1986), 285–286.
38. Li, M. and Vitányi, P. Average case complexity under the universal distribution equals worst-case complexity. *Information Processing Letters 42*, 3 (1992), 145–149.
39. McCabe-Dansted, J., Pritchard, G. and Slinko, A. Approximability of Dodgson's rule. *Social Choice and Welfare 31*, 2 (2008), 311–330.
40. Meir, R., Procaccia, A., Rosenschein, J. and Zohar, A. The complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research 33* (2008), 149–178.
41. Pennock, D., Horvitz, E. and Giles, C. Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In *Proceedings of the 17th National Conference on Artificial Intelligence* (July/Aug. 2000). AAAI Press, 729–734.
42. Procaccia, A. and Rosenschein, J. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research 28* (2007), 157–181.
43. Rothe, J., Spakowski, H. and Vogel, J. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems 36*, 4 (2003), 375–386.
44. Schöning, U. Complete sets and closeness to complexity classes. *Mathematical Systems Theory 19*, 1 (1986), 29–42.
45. Simon, H. *The Sciences of the Artificial*. MIT Press, 1969. Third edition, 1996.
46. Walsh, T. Uncertainty in preference elicitation and aggregation. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence* (July 2007). AAAI Press, 3–8.
47. Xia, L. and Conitzer, V. Generalized scoring rules and the frequency of coalitional manipulability. In *Proceedings of the 9th ACM Conference on Electronic Commerce* (July 2008). ACM Press, NY, 109–118.
48. Xia, L. and Conitzer, V. A sufficient condition for voting rules to be frequently manipulable. In *Proceedings of the 9th ACM Conference on Electronic Commerce* (July 2009). ACM Press, NY, 99–108.
49. Zuckerman, M. Procaccia, A. and Rosenschein, J. Algorithms for the coalitional manipulation problem. *Artificial Intelligence 173*, 2 (2009), 392–412.

Piotr Faliszewski (faliszew@agh.edu.pl) is an assistant professor at the AGH University of Science and Technology, Kraków, Poland.

Edith Hemaspaandra (eh@cs.rit.edu) is a professor at the Rochester Institute of Technology, Rochester, NY.

Lane A. Hemaspaandra (lane@cs.rochester.edu) is a professor at the University of Rochester, Rochester, NY.