AI's War on Manipulation: Are We Winning?

Piotr Faliszewski[*]
Department of Computer Science
AGH University of Science and Technology
Kraków, Poland
faliszew@agh.edu.pl

Ariel D. Procaccia[†]
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA
arielpro@seas.harvard.edu

Abstract

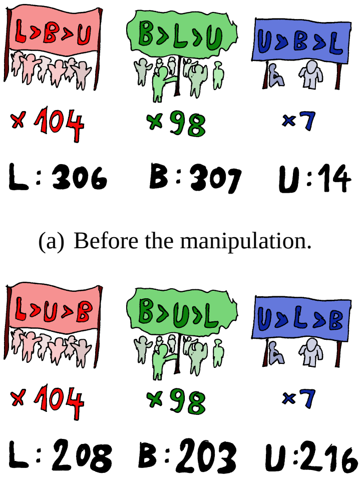
We provide an overview of more than two decades of work, mostly in AI, that studies computational complexity as a barrier against manipulation in elections.

Introduction

It was a late evening in Bitotia. The next day was going to be a big day: Citizens of Bitotia would once and for all establish which byte order was better, big-endian (B) or little-endian (L). Little Bit Timmy was a big supporter of little endian because that would give him the best position in the word. However, the population was split quite evenly between L and B, with a small minority of Bits who still remembered the single tape Turing machine and preferred unary encoding (U), without any of this endianness business. Nonetheless, about half of the Bits preferred big-endian ($B > L > U$), and about half were the other way round ($L > B > U$). The voting rule was simple enough: You gave 2 points to your top choice, 1 point to your second-best, and 0 points to the worst. As Timmy was about to fall asleep, a sudden realization struck him: Why vote $L > B > U$ and give the point to B, when U is not winning anyway? Immediately, Timmy knew: He would vote $L > U > B$!

The next day brought some of the most sensational news in the whole history of Bitotia: Unary system had won! There were 104 votes $L > U > B$, 98 votes $B > U > L$, and 7 votes $U > B > L$ (Bitotia is a surprisingly small country.) U had won with 216 points, while B had 203 and L had 208. Apparently, Timmy was not the only one who found the trick. Naturally, Bitotians wanted to find out if they could avoid such situations in the future, but... since they have to use unary now, we will have to help them!

The above story is an illustration of what we call *election manipulation*. A manipulative voter decides to cast a vote that is different from his true preferences in order to obtain a more desirable outcome. If every supporter of L voted $L > B > U$ (104 votes) and every supporter of B voted $B > L > U$ (98 votes), and the remaining 7 votes were $U > B > L$, then B would have won (see Figure 1). However, if Timmy were the only one to submit a manipulative vote, then L and B would tie for victory (and if only several more supporters of L would cast manipulative votes, L would have won). Yet, one of the dangers of manipulation is that voting rules are designed to aggregate votes accurately and if many voters attempt manipulation, the result of the election can be skewed quite significantly. In our case, the clearly least favorable option, U, ended up winning.[1]



(a) Before the manipulation.

(b) After the manipulation.

Figure 1: The election in Bitotia.

[*] Supported in part by AGH University of Science and Technology grant 11.11.120.865, by Polish Ministry of Science and Higher Education grant N-N206-378637, and by the Foundation for Polish Science's Homing/Powroty.

[†] Supported in part by a Rothschild postdoctoral fellowship.

[1] We should point out that our example is very much in spirit of *safe manipulation*, introduced by Slinko and White (2008).

Of course, part of the Bitotians' problem was that they chose an election rule—called Borda count—that seems particularly vulnerable to manipulation: It is very tempting to rank the most preferred candidate first and to rank his strongest competitor last, even if we think that the competitor is not so bad after all. They should have known better and should have picked a better voting rule! The only glitch is that there are no better voting rules: The classic result of Gibbard (1973) and Satterthwaite (1975) says that every reasonable[2] voting rule for three candidates or more sometimes creates incentives for voters to attempt manipulation.

The danger of manipulation is quite clear in human elections, but recently voting manipulation has also endangered the world of artificial intelligence and computer science. The reason is that virtual elections have become a standard tool in preference aggregation. The idea of employing voting in AI originates from the work of Ephrati and Rosenschein (1991) where elections were used to solve planning problems in multiagent systems (very briefly, the agents can vote on the next step of the plan, without revealing too much of their internal goals and desires). Another very prominent application was the web metasearch engine developed by Dwork et al. (2001). The engine treated other search engines as voters and the web pages as candidates in a virtual election. At the intersection of the worlds of computer science and human societies, voting mechanisms were used, e.g., to build recommender systems (Ghosh et al. 1999), for collaborative filtering (Pennock, Horvitz, and Giles 2000), or even to plan the development of computer systems (see the Debian project, which uses a rather advanced voting method).

The threat of manipulation is particularly relevant in the context of multiagent systems: Software agents have all the patience and computing power necessary to perform complicated analysis of elections and provide optimal manipulative votes. Additionally, they are not bound by moral obligation to act honestly as in multiagent systems their goal is to maximize their own utility (or, their owner's utility).

Manipulation is one of very many types of attack on elections. For example, in transferable utility settings, an agent may offer payments to those voters that change their votes to his liking (bribery). An agent that controls the process of voting might attempt tricks such as adding spoiler candidates (to split the votes of competitors; for example, in the U.S. 2000 presidential race it is often believed that if Ralph Nader did not participate, Al Gore would have beat George Bush), or make it difficult for some agents to cast votes. Attempts to change the result of an election via adding/deleting candidates/voters are called *election control*. Software agents can systematically plan attacks on elections using each of these types of actions (as well as many other types).

Is there any way in which we can protect elections from these attacks? Quite surprisingly, in the late 80s and early 90s, Bartholdi, Tovey, and Trick (1989; 1992) and Bartholdi and Orlin (1991) answered: *Yes! Even though manipulative actions are possible in principle, we can prevent them in practice!* They observed that even though elections are vulnerable to most types of attack, a given attack can be carried out only if it can be computed effectively. What does it mean? For example, let us consider some voting rule R and the problem of manipulation (strategic voting). Bartholdi, Orlin, Tovey, and Trick said that if given the votes of all remaining voters it is still NP-hard to compute a manipulative vote, then we can rest assured that R is *computationally* resistant to manipulation. Even if someone wanted to manipulate elections where rule R is used, short of randomly guessing the correct vote, this person would never succeed in time! Naturally, this idea of a computational barrier to manipulative behavior extends to bribery and control and to all other types of attack.

Many researchers have pursued the direction pioneered by Bartholdi, Orlin, Tovey, and Trick, studying the computational complexity of manipulation, control, and bribery in elections, obtaining results for a great number of voting rules in very varied settings; we will survey some of these results in the sequel. However, recently the computational barrier approach has also been criticized.

The most controversial part of the approach is that it relies on NP-hardness as a measure of computational difficulty. The issue is that NP-hardness is a worst-case notion and the fact that a problem is NP-hard simply means that it has *some* difficult instances and not that necessarily the ones typically occurring in practice are hard to solve. For example, let us consider the PARTITION problem, where we are given a sequence $s_1, \ldots, s_n$ of nonnegative integers and we ask if there is a subset of them that sums up to $\frac{1}{2}\sum_{i=1}^{n} s_i$. Even though the problem is NP-hard (and, in fact, NP-complete), we can solve it in polynomial time if the values $s_1, \ldots, s_n$ are sufficiently small (specifically, we can solve PARTITION in polynomial time with respect to n and $\max\{s_1, \ldots, s_n\}$). We can also effectively compute arbitrarily-close-to-correct approximate solutions to an optimization variant of the problem, where we ask for a subset whose sum is at most $\frac{1}{2}\sum_{i=1}^{n} s_i$ but as close to it as possible (though, of course, the better the approximation the longer the running time). From the practical perspective, in a large majority of settings PARTITION is easy to solve. The worry regarding com-

[2] "Reasonable" has a very natural, formal meaning here: For the Gibbard-Satterthwaite theorem the rule is reasonable if it is not dictatorial (that is, there is no special voter who chooses the winner on his own) and each candidate has a chance of winning (that is, for each candidate there is a set of votes that elect him or her). Indeed, each practically useful voting rule satisfies these conditions.

putational barrier approach is that, perhaps, theoretically hard manipulation problems are also practically easy to solve.

The main purpose of this paper is to present the results regarding manipulation in voting, both challenging the worst-case computational barrier approach and developing its theory. We believe that considering both types of results leads to a significantly better understanding of computational aspects of voting.

Elections

Let us now define our election model and describe several voting rules that we focus on in this paper.

Formally, an election $E = (C, V)$ consists of a set of candidates (or alternatives) denoted $C = \{c_1, \dots, c_m\}$ and a sequence of voters $V = (v_1, \dots, v_n)$. Each voter v_i has some preferences regarding the candidates. For example, if $C = \{a, b, c\}$ and v_1 thinks that a is the best candidate, c is second, and b is the worst, we say that v_1's preference order is $a > c > b$. There are also other means to express preference. For example, in approval voting agents simply indicate which candidates they approve of, and in range voting they assign numerical scores to candidates proportionally to the level of support. Nonetheless, preference orders are the standard model. We identify voters' preference orders with their votes.

Given the votes, a voting rule says which candidates are winners. Partially due to the Gibbard-Satterthwiate theorem, and partially due to the famous result of Arrow (1951),[3] there are remarkably many voting procedures, and new ones are still being developed (for example, Schulze method (2003), a very popular voting system used, e.g., by Wikimedia, has been developed in the late 90s). In this paper we will look just at several typical examples.

Perhaps the simplest and the most popular one is the Plurality rule: In Plurality we simply give each candidate one point for each vote that ranks him first, and we declare as winners those candidates that have most points. Note that we *do* allow multiple winners. In practice, elections involve various tie-breaking rules, but here (and typically in the CS literature) we disregard such complications, and instead use one of the following models. In the *unique-winner model*, a candidate has to be the unique winner to claim victory in the election, and in the *nonunique-winner model* it suffices that the candidate is one of the winners.

Plurality rule is the simplest member of a class of election systems called *(positional) scoring rules*. A scoring rule for m candidates is defined by a vector $\alpha = (\alpha_1, \dots, \alpha_m)$ of nonnegative integers such that $\alpha_1 \geq \cdots \geq \alpha_m$. A candidate receives α_j points for each vote that ranks him on the jth position; the winners are those candidates who get most points. It is easy to see that Plurality is defined via a family of scoring rules $(1, 0, \dots, 0)$, with one vector for each number of candidates. Similarly, Borda count—the rule used by Bitotians—is defined via a family of scoring vectors of the form $(m-1, m-2, \dots, 0)$. A scoring rule is used, e.g., for the elections of the best song in the Eurovision song contest.

Copeland's rule presents a very different perspective on choosing a winner. Let a and b be two candidates in an election E. We say that a wins a head-to-head contest with b if the majority of voters prefers a to b. In Copeland's rule a candidate receives one point for each candidate that he defeats in a head-to-head contest, and half a point for each candidate with whom he ties. That is, Copeland's rule views the process of electing the winner as a round-robin tournament, with 1 point for victory, $\frac{1}{2}$ point for a tie, and 0 points for losing. The candidates with most points are winners. Sometimes, instead of using half-points for a tie, a different value $\alpha \in [0, 1]$ is used, and the voting rule is denoted Copeland$^\alpha$ (Faliszewski et al. 2009b) (though, we mention that some papers also use the term Copeland for what we would call Copeland^0).

Manipulation and Related Problems

To formally study computational properties of manipulation we have to define it as a decision problem. We will do so in this section, discussing several variants of the definition and several related problems.

Let R be a voting rule. Intuitively, in the R-MANIPULATION problem we are given an election where some of the voters have fixed votes (preference orders) and some voters—the manipulators—are trying to choose their votes so that their preferred candidate p becomes a winner. The manipulators are working together, i.e., they form a coalition, and we assume that they can perfectly coordinate their actions. We also assume that they have perfect knowledge regarding the remaining votes. These assumptions stem from the fact that we are interested in hardness of manipulation in a setting that is most favorable for the manipulators. If the manipulation is hard there then certainly it must be hard in more realistic settings.[4] Formally, we have the following definition (based on that from (Bartholdi, Tovey, and Trick 1989) and (Conitzer, Sandholm, and Lang 2007)).

Definition 1. *Let R be a voting rule. In R-*MANIPULATION *we are given an election $E = (C, V +$*

[3] Arrow's impossibility theorem gives several very natural requirements that an intuitively good voting rule should satisfy and shows that there are *no* voting rules that satisfy all of them. As a result, what voting rule is best depends on the setting and hence there are multiple different ones to choose from.

[4] One should be careful here: In a less favorable setting the goals of the manipulators might also be less demanding.

W), where voters in V have fixed preference orders and the preference orders of voters in W remain to be set, and a designated candidate $p \in C$; we ask if there is a way to set the votes in W so that p is a winner.

Originally, Bartholdi, Tovey, and Trick (1989) considered *single-voter* manipulation instances only, that is, those where the collection W contains exactly one voter. The definition presented here—adapted from (Conitzer, Sandholm, and Lang 2007)—regards *coalitional* manipulation. In fact, Conitzer, Sandholm, and Lang introduced one more important twist to the definition: They allowed voters to be weighted. In R-WEIGHTED-MANIPULATION each voter v (manipulator or not) has a *weight* w_v and his vote counts as w_v votes. Weighted elections are very natural. For example, within a company, the votes of shareholders are weighted by the amount of shares they hold, the U.S. electoral college is weighted, and so are countries voting within European Union.

Finally, a different variant of the manipulation problem was studied by Meir et al. (2008), who considered multiwinner elections (such as, e.g., elections for assemblies or parliaments).

Manipulation captures situations where a group of voters, the coalition, decides to collude in order to obtain a better outcome for itself. On the other hand, in bribery there is a single agent who wishes to change the outcome of the election and offers payments to voters for changing the preference orders to his liking. The computational study of bribery was initiated by Faliszewski, Hemaspaandra, and Hemaspaandra (2009) who, in particular, introduced and studied the following problem.

Definition 2. *Let R be a voting rule. In the R-*BRIBERY *problem we are given an election $E = (C, V)$, a designated candidate $p \in C$, and a natural number B. We ask if it is possible to ensure that p is an R-winner of E via changing the votes of at most B voters.*

As in the case of manipulation, we can consider the weighted variant of the problem, R-WEIGHTED-BRIBERY. However, in the case of bribery, perhaps a different twist of the definition is more interesting. In R-BRIBERY, effectively, each voter has the same unit cost: We only care about bribing as few voters as possible. However, in many settings, the voters might have different prices, depending, e.g., on how much a particular voter cares about the result of the election or on the nature of the bribery. To model the first possibility, Faliszewski, Hemaspaandra, and Hemaspaandra (2009) introduced R-\$BRIBERY where each voter v has a price π_v for changing his vote (after we pay v the π_v units, we obtain full control over v's vote). To deal with the latter option, Elkind, Faliszewski, and Slinko (2009) introduced R-SWAP-BRIBERY. In swap bribery each voter v has a cost function π_v such that for each two candidate c, c', $\pi_v(c, c')$ is the cost of swapping c and c' on v's preference list (provided c and c' are ranked next to each other). For example, a voter might be willing to swap his two least favorite candidates at a small cost, but would never—irrespective of the payment—change the top-ranked candidate. The goal of the briber is to find a sequence of adjacent swaps that lead to his or her preferred candidate's victory, and that has lowest cost.

The priced variants of the bribery problem can also be considered in the weighted setting. However, essentially, all such problems are NP-complete as this is true even with respect to Plurality-WEIGHTED-\$BRIBERY.

We will not survey results regarding bribery in this paper and we point the reader to particular research articles. However, the general intuition is that bribery appears to be computationally harder than manipulation. This intuition is based on the results for *natural* systems studied so far. However, there is an artificial election system for which manipulation is NP-complete but bribery is in P (Faliszewski, Hemaspaandra, and Hemaspaandra 2009).

There is one more problem that is quite related to manipulation and bribery, namely the *Possible-Winner* problem introduced by Konczak and Lang (2005) and further studied by, e.g., (Xia and Conitzer 2008a; Walsh 2007; Pini et al. 2007; Betzler, Hemmann, and Niedermeier 2009). Let us fix a voting rule R. In R-POSSIBLE-WINNER we are given an election $E = (C, V)$ where the preference orders are possibly partial (a partial order is, simply, a reflexive, transitive, antisymmetric relation). The question is: Given a candidate p, is it possible to extend the preference orders to complete linear orders so that p is a winner? The possible winner problem models a situation where we have some partial information about the votes and we want to find out who still has a chance of winning. Similarly, in the R-NECESSARY-WINNER problem—also defined in (Konczak and Lang 2005)—we ask if a given candidate is a winner irrespective of how the votes are completed.

Formally, manipulation is a special case of the possible winner problem, where the nonmanipulators have fully specified preference orders and the manipulators have completely unspecified ones. In fact, quite a few problems mentioned above are special cases of each other. For example R-MANIPULATION is a special case of R-\$BRIBERY (we can view the manipulation problem as a bribery problem where the prices of manipulators are very low, the prices of nonmanipulators are very high, and our budget allows us to buy the votes of all the manipulators, but none of the nonmanipulators). It is somewhat less trivial to see that R-POSSIBLE-WINNER is a special case of R-SWAP-BRIBERY. We present the "is a special case of" relations between problems in Figure 2. These relations are important as computational hardness of a special case implies hardness of the more

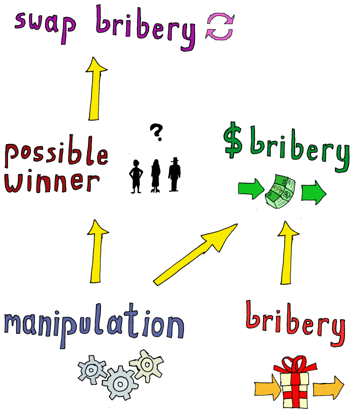


Figure 2: Diagram of “is a special case of” relation for manipulation-related problems (R is a voting rule). An arrow points from a problem that is a special case to the problem that generalizes it. The same diagram is true for the weighted variants of the problems.

general problem, and easiness of the more general problem implies easiness of its special cases. For pairs of problems in Figure 2 for which we do not indicate “is a special case of” relation, either such a relation does not hold or is not known to hold. (Figure 2 presents results from (Faliszewski, Hemaspaandra, and Hemaspaandra 2009; Elkind, Faliszewski, and Slinko 2009).)

As a final remark, we mention that researchers often consider destructive variants of the problems we have presented here, where the goal is to ensure that some candidate does not win. However, in this paper we focus on the constructive cases only (as presented in the definitions in this section).

Hardness of Manipulation

With all the necessary background, we can finally move on to the discussion of computational aspects of manipulation. In sync with history, we start with the single-manipulator variant of the problem.

Bartholdi, Tovey, and Trick wanted to show hardness of manipulation, but their first result was, in fact, that for a large class of voting rules, including all scoring rules and Copeland, single-voter manipulation is easy. The manipulator has to execute the following natural steps:

Initialization: Place the preferred candidate p in the first position in the vote.

Loop: If there are no more unprocessed candidates, we have found a successful manipulative vote. Otherwise, test if there is a not-yet-placed candidate c such that putting c in the next free position in the vote does not prevent p from being a winner. If there is such a c, place him in the vote. Otherwise, signal that manipulation is impossible. Repeat.

It is quite easy to see that this algorithm works both for Copeland and for each scoring rule. Placing p in the first position in the vote completely determines p’s score, and the order in which we fill in positions in the vote (from the most preferred to the least preferred) guarantees that each time we place a new candidate we can also determine his final score.

Given the naturalness and simplicity of the above algorithm, it is in fact quite remarkable that Bartholdi, Tovey, and Trick (1989) and Bartholdi and Orlin (1991) have actually found voting rules for which single-voter manipulation is NP-hard. These rules are, respectively, second-order Copeland (a variant of the Copeland rule with an elaborate tie-breaking) and a variant of single transferable vote (STV). Very briefly speaking, STV works as follows: If there is a single candidate, he is the winner; otherwise find a candidate that is ranked first the least number of times, remove him from the votes, and repeat. STV is quite vulnerable to internal tie-resolutions (that is, the order in which candidates with the same number of first-place votes are removed). In fact, a recent result shows that for a certain natural tie-breaking rule even determining the winners in STV can become NP-complete (Conitzer, Rognlie, and Xia 2009).

Second-order Copeland and STV were the first rules for which computational resistance to manipulation was obtained. In fact, to date, only one more natural voting rule—called Ranked Pairs—is known to posses such resistance to single-voter manipulation (Xia et al. 2009). However, Conitzer and Sandholm (2003) showed how adding a preround can make single-voter manipulation computationally hard, and Elkind and Lipmaa (2005a) achieved the same by building hybrid election systems, in a way resembling STV.

The early results of Bartholdi, Orlin, Tovey, and Trick (almost) exhaust the research regarding single voter manipulation; their greedy algorithm is indeed very powerful. Let us, thus, turn to coalitional manipulation. There are two main flavors of the problem: weighted and unweighted. Historically, weighted manipulation has been studied earlier (and much more thoroughly) so let us consider it first.

Weighted manipulation was introduced by Conitzer, Sandholm, and Lang (2007) who observed that in real life elections (that is, in elections we encounter in human societies) typically there are very few candidates. They have pointed out that if the number of candidates is a small constant then (unweighted) manipulation immediately becomes easy—one can, essentially, brute-force through all possible combinations of votes (if there are m candidates and n manipulators then there are at most $(n+1)^{m!}$ different combinations of manipulative votes—assuming the order of votes is irrelevant—

and if m is a small constant then, at least in principle, we can look at each set of votes for the manipulators). However, in weighted elections the brute-force approach does not work anymore. Even if there are very few candidates, it is not sufficient to know how many votes of each type there are, but it is critical to know which voters have which preference orders. Thus, Conitzer Sandholm, and Lang (2007) set out to determine the complexity of weighted manipulation for a number of voting rules (they have considered nearly a dozen rules including Plurality, Veto, Borda, Copeland). In fact, not only have they done that, but also for each rule they have established the exact number of candidates that need to participate in the election for the weighted manipulation problem to be NP-complete. It turned out that for most rules, as soon as we have at least 3 or 4 candidates, weighed manipulation becomes NP-complete. However, there are some quite interesting rules (e.g., so-called *randomized cup*) for which weighted manipulation is in P for up to 6 candidates and then—suddenly—becomes NP-complete if there are 7 candidates or more.

Perhaps the most beautiful result regarding the complexity of weighted manipulation is the dichotomy theorem of Hemaspaandra and Hemaspaandra (2007), which classifies the complexity of weighted manipulation in scoring rules: Given a scoring rule $\alpha = (\alpha_1, \ldots, \alpha_m)$, weighted manipulation is NP-complete if α satisfies the *diversity of dislike* condition, that is $\{\alpha_2, \ldots, \alpha_m\}$ contains at least two values. Otherwise, weighted manipulation is in P. The proof of Hemaspaandra and Hemaspaandra relies on the fact that the voters' votes are not restricted in any way; any voter can cast any possible vote. However, if one does restrict possible votes—e.g., via assuming that the electorate is single-peaked—then the dichotomy condition changes. Recently, Faliszewski et al. (2009a), showed a variant of the dichotomy for scoring protocols with 3 candidates for single-peaked electorates. Study of manipulation under single-peaked electorates, initiated by Walsh (2007), is a very interesting direction of research as single-peaked preferences often arise in practice. (Single-peaked preferences, introduced by Black (1958), model situations where voters judge candidates based on their view on a single issue such as, e.g., level of taxation.)

Compared to weighted coalitional manipulation, surprisingly little is known about the unweighted case. There is only a handful of voting rules for which the complexity of unweighted coalitional manipulation has been determined. Zuckerman, Procaccia and Rosenschein (2009) showed that the problem is easy for Veto and for a voting rule called Plurality with runoff, Faliszewski, Hemaspaandra, and Schnoor (2008; 2010) showed hardness for Copeland$^{\alpha}$ (for $\alpha \in [0,1]-\{0.5\}$), and Xia et al. (2009) showed hardness for Maximin and Ranked Pairs, and easiness for Bucklin. It is quite interesting that for all of these rules for which coalitional manipulation is hard—but single voter manipulation is easy—it suffices that we have exactly *two* manipulators to reach hardness. That is, even the need to coordinate such a small coalition is enough to boost the complexity of these problem to NP-completeness.

Some earlier results regarding unweighted manipulation include those of Elkind and Lipmaa (2005b) (they used one-way functions to tweak voting rules so that the resulting rules are computationally resistant to unweighted manipulation) and of Conitzer, Sandholm, and Lang (2007), who analyzed connections between weighted manipulation and unweighted manipulation for the case where votes are not known with certainty.

Unfortunately, so far, no result resembling the dichotomy theorem for unweighted manipulation under (polynomial-time computable families of) scoring rules is known, and it appears that obtaining one may be very difficult.[5] Very recently, Xia et al. (2010) established that there is a polynomial-time computable family of scoring rules where unweighted manipulation is NP-complete. However, to date even the exact complexity of unweighted manipulation for Borda is not known (though, see the next section for a discussion). We believe that establishing such a result is a very interesting, difficult challenge and we very much hope that some of the readers of this paper will successfully tackle it!

Challenging the Worst-Case Approach

The previous section surveyed a significant body of work devoted to variations on the following theme: preventing manipulation via computational complexity. The results provide a rather rich understanding of the intricate dependence between the characteristics of the voting rule used to govern the election, and the computational complexity of manipulating it. However, these results are all concerned with *worst-case* hardness. As we mentioned above, it may still be the case that voters are *usually* able to manipulate the election even though the voting rule in question is worst-case hard to manipulate. In this section we survey the literature that challenges the worst-case approach by asking: Is there a reasonable voting rule that is usually hard to manipulate? In the sequel we discuss three approaches to answering this question (in the negative!).

The "Window of Error" approach

Although what one means by "a *reasonable* voting rule" may be arguable, the main difficulty in answering the above question is that it is unclear what one means by

[5] Note that if the number of candidates is fixed then manipulation under *any* scoring protocol is easy. Thus, we ask here for a dichotomy theorem regarding *families* of scoring protocols.

"usually". Ideally, we would like the results to hold under any plausible distribution on the votes, but it is *a priori* unclear which formal methodology can achieve such an ambitious goal.

The first to tackle these rather intimidating issues were Procaccia and Rosenschein (2007b). They presented the notion of *junta distributions*; very generally speaking, these are distributions over the instances of R-MANIPULATION that satisfy several constraints. Procaccia and Rosenschein then informally argued that a junta distribution may possess the following property: if an algorithm often succeeds in deciding R-MANIPULATION when the instances of the problem are distributed according to a junta distribution, it would also succeed in deciding R-MANIPULATION when the instances are distributed according to many other plausible distributions. Procaccia and Rosenschein presented a greedy algorithm that often succeeds in deciding R-MANIPULATION, where R is a scoring rule, and the instances are distributed with respect to a specific distribution that is proven to be a junta distribution. This may provide some evidence that *scoring rules are usually easy to manipulate*. However, other authors have argued that the notion of junta distributions has limited usefulness (Erdélyi et al. 2009).

In retrospect, the crux of the paper of Procaccia and Rosenschein (2007b) is a rather loose characterization of instances on which the greedy algorithm may fail; these instances are drawn with small probability according to their junta distribution. More recently, Zuckerman et al. (2009) significantly refined this idea. In particular, by obtaining a more careful characterization of the greedy algorithm's hard instances, they show that the greedy algorithm of Procaccia and Rosenschein has the following property with respect to Borda: Given a "yes" instance of R-MANIPULATION with a set W of manipulators, the algorithm may wrongly return a negative answer, but would in fact find a successful manipulation given $|W| + 1$ manipulators. (On the other hand, if the algorithm answers "yes", the answer is certainly correct.) It is possible to define an optimization problem whose solution is the minimum number of unweighted manipulators needed to make a given candidate win; then the greedy algorithm approximates the solution to this problem under Borda to an additive term of one. The intuitive implication is that the "size" of the algorithm's "window of error" is one manipulator, which in turn implies that the algorithm would succeed with high probability under many distributions.

A slightly weaker extension of the above result to scoring rules in general was obtained by Xia et al. (2010). In another related paper Brelsford et al. (2008) set up a general framework for studying approximation in manipulation and other problems; as a corollary of their main theorem they obtain a version of the above result of Zuckerman et al. that holds for a large subset of scoring rules but requires that the number of candidates is constant.

In general, the abovementioned papers (except the one by Brelsford et al.) design algorithms that are "usually" able to manipulate certain voting rules, by arguing that these algorithms fail on very specific instances. The drawback of this approach is that the algorithms are tailor-made for the voting rules in question (scoring rules, Maximin, Plurality with Runoff), and hence this approach cannot give a completely satisfying answer to the question posed at the beginning of the section.

The "Fraction of Manipulators" approach

In a different paper, Procaccia and Rosenschein (2007a) made the following observation, which is an extension of similar results in the social choice literature (Baharad and Neeman 2002; Slinko 2004): If the number of manipulators is large then there almost always exists a successful manipulation, whereas if the number of manipulators is small then there almost always does not exist a successful manipulation. Specifically, they consider scoring rules, and show that if $|W| = \omega(\sqrt{|V|})$ then there exists a trivial manipulation[6] with high probability, and if $|W| = o(\sqrt{|V|})$ then there does not exist a manipulation with high probability. This result holds under rather general assumptions on the distribution over the given votes in V. Moreover, the above result was generalized by Xia and Conitzer (2008b); their theorems hold for *generalized scoring rules*, a large class of voting rules that includes the voting rules mentioned above (scoring rules, Copeland, STV, Ranked Pairs), as well as other commonly studied voting rules. These results suggest that in the vast majority of cases the R-MANIPULATION problem can be solved efficiently under generalized scoring rules.

A gap that the foregoing papers left open is the case where $|W| = \Theta(\sqrt{|V|})$; this case seems unwieldy as far as analytical analysis is concerned. Walsh (2009) recently addressed this difficulty using an empirical methodology. In particular, analyzing the Veto rule, he demonstrates that there is a smooth transition, from nearly zero to nearly one, in the probability that there exists a successful manipulation when the size of the manipulating coalition grows.

Note that, although the class of generalized scoring rules certainly includes many natural voting rules, it is still not wide enough to preclude the existence of a reasonable voting rule that is usually hard to manipulate. Indeed, in more recent work, Xia and Conitzer (2009) characterized this class using two axiomatic properties, *anonymity* (indifference to the identities of the voters) and a new axiom called *finite local consistency*. Their

[6] The manipulators rank their preferred candidate p first, and every other candidate is ranked last by roughly $|W|/(|C| - 1)$ manipulators.

characterization implies that the well-studied Dodgson rule (see, e.g., (Homan and Hemaspaandra 2006; Caragiannis et al. 2009)) is not a generalized scoring rule.

The axiomatic approach

Generally speaking, the last set of papers that we wish to discuss makes the following argument: all reasonable voting rules satisfy some axioms, and all the voting rules satisfying said axioms are usually manipulable by a trivial algorithm.

The first to take this approach were Conitzer and Sandholm (2006). They showed that an R-MANIPULATION instance can be decided easily if it satisfies two properties: weak monotonicity, which is a very natural property, and the more controversial property that the manipulators can make one of exactly two candidates win the election. Conitzer and Sandholm empirically demonstrated that the latter property holds with high probability under different voting rules, but their simulations were carried out only with respect to specific distributions and a small number of candidates.

More recently, Friedgut et al. (2008) proposed a promising line of attack that does not impose stringent restrictions on the voting rule. They assumed that the distribution over votes is uniform, that is, we draw a uniformly random ranking independently for each voter; this assumption is known as the *impartial culture assumption* in the social choice literature. Friedgut et al. also assume that there is a single manipulator. Their main insight is that a completely random manipulation may succeed with non-negligible probability.[7] In more detail, consider a trivial randomized algorithm that, given the preferences of the voters, chooses a random ranking as the strategic vote of the manipulator; if this strategy provides a successful manipulation with nonnegligible probability on a given instance, then by repeating this procedure we can achieve a high probability of success with respect to that instance. The main result of Friedgut et al. is, roughly speaking, as follows. Assume there are exactly three candidates, and the voting rule is *neutral*, i.e., the outcome is independent of the names of the candidates. If the trivial randomized algorithm succeeds with only negligible probability when a preference profile and a manipulation are drawn uniformly at random, then the voting rule must be very close to being a dictatorship, in the sense that there is one voter such that his favorite candidate is almost always selected by the voting rule. The appeal of this result is that one can easily argue that it indeed captures every reasonable voting rule. However, its impact is limited by the fact that it only holds for a restricted number of candidates and under the impartial culture assumption.

Several attempts have been made to extend the above result. Xia and Conitzer (2008c) achieved a similar result that holds for any constant number of candidates, albeit requires more restrictive assumptions on the voting rule. Dobzinski and Procaccia (2008) established an analogous result for the case of two voters and any number of candidates, under a comparably weak assumption on the voting rule. Very recently the result of Friedgut et al. was successfully extended to settings with an arbitrary number of voters and candidates, in an impressive demonstration of mathematical prowess due to Isaksson et al. (2010).[8]

The last result settles in the negative the question of the existence of voting rules that are usually hard to manipulate, as long as one is willing to accept the impartial culture assumption. Nevertheless, it is still possible to argue that in most settings, both in the context of political elections and multiagent domains, the votes tend to exhibit structure that is far from random (the work of Walsh (2007) and of Faliszewski et al. (2009b) on manipulating single-peaked elections is an example of a step in that direction, albeit in the worst-case complexity model). Therefore, the final word regarding the (non)existence of voting rules that are usually hard to manipulate is yet to be said.

Summary

In the first part of the survey we discussed worst-case hardness as a barrier against manipulation in elections. The results along this line of work show that several formulations of the manipulation problem are computationally hard under different voting rules. After more than two decades of research we have a deep understanding of the worst-case complexity of manipulation in elections. An enigmatic open problem is the complexity of unweighted manipulation under Borda.

In the second part of the survey we outlined three lines of work that challenge the worst-case approach. Ideally, one would like to design a reasonable voting rule that is "usually" hard to manipulate. Unfortunately, to date all the work in this direction suggests that there is no such voting rule. However, despite significant progress over the last few years, this issue has not yet been settled decisively, and still gives rise to fascinating methodological and mathematical questions.

[7] This is trivial under the formulation of the manipulation problem given in Definition 2 when p is chosen at random. Friedgut et al. consider a slightly different, in a sense more focused, formulation of the problem, where the manipulator also holds a ranking and the question is whether he can vote in a way that improves the outcome according to his preferences.

[8] Isaksson et al. (2010) consider manipulations where four adjacent candidates are randomly permuted.

References

Arrow, K. 1951. *Social Choice and Individual Values*. John Wiley and Sons.

Baharad, E., and Neeman, Z. 2002. The asymptotic strategyproofness of scoring and Condorcet consistent rules. *Review of Economic Design* 4:331–340.

Bartholdi, J., and Orlin, J. 1991. Single Transferable Vote resists strategic voting. *Social Choice and Welfare* 8:341–354.

Bartholdi, J.; Tovey, C. A.; and Trick, M. A. 1989. The computational difficulty of manipulating an election. *Social Choice and Welfare* 6:227–241.

Bartholdi, J.; Tovey, C. A.; and Trick, M. A. 1992. How hard is it to control an election. *Mathematical and Computer Modelling* 16:27–40.

Betzler, N.; Hemmann, S.; and Niedermeier, R. 2009. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proc. of 21st IJCAI*, 53–58.

Black, D. 1958. *Theory of Committees and Elections*. Cambridge University Press.

Brelsford, E.; Faliszewski, P.; Hemaspaandra, E.; Schnoor, H.; and Schnoor, I. 2008. Approximability of manipulating elections. In *Proc. of 23rd AAAI*, 44–49.

Caragiannis, I.; Covey, J. A.; Feldman, M.; Homan, C. M.; Kaklamanis, C.; Karanikolas, N.; Procaccia, A. D.; and Rosenschein, J. S. 2009. On the approximability of Dodgson and Young elections. In *Proc. of 20th SODA*, 1058–1067.

Conitzer, V., and Sandholm, T. 2003. Universal voting protocol tweaks to make manipulation hard. In *Proc. of 18th IJCAI*, 781–788.

Conitzer, V., and Sandholm, T. 2006. Nonexistence of voting rules that are usually hard to manipulate. In *Proc. of 21st AAAI*, 627–634.

Conitzer, V.; Rognlie, M.; and Xia, L. 2009. Preference functions that score rankings and maximum likelihood estimation. In *Proc. of 21st IJCAI*, 109–115.

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

Dobzinski, S., and Procaccia, A. D. 2008. Frequent manipulability of elections: The case of two voters. In *Proc. of 4th WINE*, 653–664.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the web. In *Proc. of 10th WWW*, 613–622.

Elkind, E., and Lipmaa, H. 2005a. Hybrid voting protocols and hardness of manipulation. In *Proc. of 16th ISAAC*, 206–215.

Elkind, E., and Lipmaa, H. 2005b. Small coalitions cannot manipulate voting. In *Financial Cryptography and Data Security*, volume 3570 of *LNCS*. Springer-Verlag. 285–297.

Elkind, E.; Faliszewski, P.; and Slinko, A. 2009. Swap bribery. In *Proc. of 2nd SAGT*, 299–310.

Ephrati, E., and Rosenschein, J. S. 1991. The Clarke Tax as a consensus mechanism among automated agents. In *Proc. of 9th AAAI*, 173–178. Journal version appeared in AMAI 20:13–67, 1997.

Erdélyi, G.; Hemaspaandra, L. A.; Rothe, J.; and Spakowski, H. 2009. Generalized juntas and NP-hard sets. *Theoretical Computer Science* 410(38–40):3995–4000.

Faliszewski, P.; Hemaspaandra, E.; ; and Schnoor, H. 2008. Copeland voting: Ties matter. In *Proc. of 7th AAMAS*, 983–990.

Faliszewski, P.; Hemaspaandra, E.; Hemaspaandra, L.; and Rothe, J. 2009a. The shield that never was: Societies with single-peaked preferences are more open to manipulation and control. In *Proc. of 12th TARK*, 118–127.

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

Faliszewski, P.; Hemaspaandra, E.; and Hemaspaandra, L. 2009. How hard is bribery in elections? *Journal of Artificial Intelligence Research* 35:485–532.

Faliszewski, P.; Hemaspaandra, E.; and Schnoor, H. 2010. Manipulation of Copeland elections. In *Proc. of 9th AAMAS*, 367–374.

Friedgut, E.; Kalai, G.; and Nisan, N. 2008. Elections can be manipulated often. In *Proc. of 49th FOCS*, 243–249.

Ghosh, S.; Mundhe, M.; Hernandez, K.; and Sen, S. 1999. Voting for movies: The anatomy of a recommender system. In *Proc. of 3rd AGENTS*, 434–435.

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

Hemaspaandra, E., and Hemaspaandra, L. A. 2007. Dichotomy for voting systems. *Journal of Computer and System Sciences* 73(1):73–83.

Homan, C., and Hemaspaandra, L. A. 2006. Guarantees for the success frequency of an algorithm for finding Dodgson election winners. In *Proc. of 31st MFCS*, 528–539.

Isaksson, M.; Kindler, G.; and Mossel, E. 2010. The geometry of manipulation – a quantitative proof of the Gibbard Satterthwaite Theorem. In *Proc. of 51st FOCS*. To appear.

Konczak, K., and Lang, J. 2005. Voting procedures with incomplete preferences. In *Proc. of 2nd M-PREF*.

Meir, R.; Procaccia, A. D.; Rosenschein, J. S.; and Zohar, A. 2008. Complexity of strategic behavior in multi-winner elections. *Journal of Artificial Intelligence Research* 33:149–178.

Pennock, D.; Horvitz, E.; and Giles, L. 2000. Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In *Proc. of 17th AAAI*, 729–734.

Pini, M. S.; Rossi, F.; Venable, K. B.; and Walsh, T. 2007. Incompleteness and incomparability in preference aggregation. In *Proc. of 20th IJCAI*, 1464–1469.

Procaccia, A. D., and Rosenschein, J. S. 2007a. Average-case tractability of manipulation in elections via the fraction of manipulators. In *Proc. of 6th AAMAS*, 718–720.

Procaccia, A. D., and Rosenschein, J. S. 2007b. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research* 28:157–181.

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

Schulze, M. 2003. A new monotonic and clone-independent single-winner election method. *Voting Matters* 17:9–19.

Slinko, A., and White, S. 2008. Is it ever safe to vote strategically? Technical Report N-563, Department of Mathematics, University of Auckland.

Slinko, A. 2004. How large should a coalition be to manipulate an election? *Mathematical Social Sciences* 47(3):289–293.

Walsh, T. 2007. Uncertainty in preference elicitation and aggregation. In *Proc. of 22nd AAAI*, 3–8.

Walsh, T. 2009. Where are the really hard manipulation problems? The phase transition in manipulating the Veto rule. In *Proc. of 21st IJCAI*, 324–329.

Xia, L., and Conitzer, V. 2008a. Determining possible and necessary winners under common voting rules given partial orders. In *Proc. of 23rd AAAI*, 196–201.

Xia, L., and Conitzer, V. 2008b. Generalized Scoring Rules and the frequency of coalitional manipulability. In *Proc. of 9th EC*, 109–118.

Xia, L., and Conitzer, V. 2008c. A sufficient condition for voting rules to be frequently manipulable. In *Proc. of 9th EC*, 99–108.

Xia, L., and Conitzer, V. 2009. Finite local consistency characterizes generalized scoring rules. In *Proc. of 21st IJCAI*, 336–341.

Xia, L.; Zuckerman, M.; Procaccia, A. D.; Conitzer, V.; and Rosenschein, J. S. 2009. Complexity of unweighted coalitional manipulation under some common voting rules. In *Proc. of 21st IJCAI*, 348–353.

Xia, L.; Conitzer, V.; and Procaccia, A. D. 2010. A scheduling approach to coalitional manipulation. In *Proc. of 11th EC*, 275–284.

Zuckerman, M.; Procaccia, A. D.; and Rosenschein, J. S. 2009. Algorithms for the coalitional manipulation problem. *Artificial Intelligence* 173(2):392–412.