Cloning in Elections

Edith Elkind[*]
School of Physical and Mathematical Sciences
Nanyang Technological University
Singapore

Piotr Faliszewski[†]
Department of Computer Science
AGH Univ. of Science and Technology
Kraków, Poland

Arkadii Slinko
Department of Mathematics
University of Auckland
Auckland, New Zealand

Abstract

We consider the problem of manipulating elections via cloning candidates. In our model, a manipulator can replace each candidate c by one or more *clones*, i.e., new candidates that are so similar to c that each voter simply replaces c in his vote with the block of c's clones. The outcome of the resulting election may then depend on how each voter orders the clones within the block. We formalize what it means for a cloning manipulation to be successful (which turns out to be a surprisingly delicate issue), and, for a number of prominent voting rules, characterize the preference profiles for which a successful cloning manipulation exists. We also consider the model where there is a cost associated with producing each clone, and study the complexity of finding a minimum-cost cloning manipulation. Finally, we compare cloning with the related problem of control via adding candidates.

Introduction

In real-life elections with more then two candidates, the winner does not always have broad political support. This is possible, for example, when the opposing views are represented by several relatively similar candidates, and therefore the vote in favor of these views gets "split". For example, it is widely believed that in the 2000 U.S. Presidential election spoiler candidate Ralph Nader have split votes away from Democratic candidate Al Gore allowing Republican candidate George W. Bush to win.

One can also imagine scenarios where having several similar candidates may bias the outcome in their favor. For example, suppose that an electronics website runs a competition for the best digital camera by asking consumers to vote for their two favorite models from a given list. If the list contains one model of each brand, and half of the consumers prefer Sony to Nikon to Kodak, while the remaining consumers prefer Kodak to Nikon to Sony, then Nikon will win the competition. On the other hand, if each brand is represented by several similar models, then the "Sony" customers are likely to vote for two models of Sony, the "Kodak" customers are likely to vote for two models of Kodak, and Nikon receives no votes.

The above-described scenarios present an opportunity for a party that is interested in manipulating the outcome of an election. Such a party—most likely, a campaign manager for one of the candidates—may invest in creating "clones" of one or more candidates in order to make its most preferred candidate (or one of its "clones") win the election. A natural question, then, is which voting rules are resistant to such manipulation, and whether the manipulator can compute the optimal cloning strategy for a given election.

A variant of this question was first studied by Tideman (1987) (see also (Laffond, Laine, and Laslier 1996)), who introduced the concept of "independence of clones" as a criterion for voting rules. He considered a number of well-known voting rules, and discovered that among these rules, STV is the only one that satisfies this criterion. However, STV does not satisfy many other important criteria for voting rules, such as Condorcet consistency. Thus, Tideman (1987) proposed a voting rule, the "ranked pairs rule," that was both Condorcet-consistent and independent of clones in all but a small fraction of settings. Subsequently, Tideman and Zavist (1989) proposed a modification of this rule that is completely independent of clones. Later it was shown that some other voting rules, such as Schulze's rule (Schulze 2003), are also resistant to cloning.

However, the existing work on cloning does not place any restrictions on the number or identities of the candidates that can be cloned, or the number of clones that can be produced. On the other hand, it is clear that in practical campaign management scenarios these issues cannot be ignored: not all candidates can be cloned, and creating a clone of a given candidate may be costly. Further, while previous work demonstrates that many well-known voting rules are susceptible to cloning, no attempt has been made to characterize the elections in which a specific candidate can be made a winner with respect to a given voting rule by means of cloning. Finally, to the best of our knowledge, there has been no work on computational aspects of cloning.

In this paper, we study feasibility and complexity of cloning viewed as a campaign management tool. We start by presenting a formal description of the manipulator's

[*]Supported by NRF Research Fellowship (NRF-RF2009-08).
[†]Supported by AGH University of Technology Grant no. 11.11.120.865, by Polish Ministry of Science and Higher Education grant N-N206-378637, and by Foundation for Polish Science's program Homing/Powroty.

cloning problem. Compared to the model of (Tideman 1987; Zavist and Tideman 1989): it has two additional components: the definition of what it means for cloning to succeed (which turns out to be a very delicate issue), and the notion of cloning costs. We then investigate the complexity of the manipulator's problem for several well-known voting rules. Interestingly, for many of these rules, the manipulable profiles can be characterized in terms of well-known notions of social choice, such as Pareto optimality, Condorcet loser, or uncovered set. However, we show that for many rules finding a minimum-cost manipulation is computationally hard, even for very simple cost models. We conclude the paper by exploring the relationship between cloning and control by adding candidates. Due to space constraints, most proofs are omitted.

Preliminaries

Given a set A of *alternatives* (also called *candidates*), a voter's *preference* R is a *linear order* over A, i.e., a total transitive antisymmetric binary relation over A. An *election* E with n voters is given by its set of alternatives A and a *preference profile* $\mathcal{R} = (R_1, \dots, R_n)$, where R_i is the preference of voter i; we write $E = (A, \mathcal{R})$. For readability, we sometimes write $\succ_i$ in place of R_i. Also, we denote by $|\mathcal{R}|$ the number of voters in $\mathcal{R}$.

A *voting rule* $\mathcal{F}$ is a mapping from pairs of the form $(A, \mathcal{R})$, where A is some finite set and $\mathcal{R}$ is a preference profile over A, to subsets of A. The elements of $\mathcal{F}(E)$ are called the *winners* of the election E. (Thus, we allow an election to have more than one winner, i.e., we work in the so-called *non-unique winner* model.) We consider the following voting rules (for all rules described in terms of scores the winners are the alternatives with the maximum score):

Plurality. The *Plurality score* $Sc_P(c)$ of a candidate $c \in A$ is the number of voters that rank c first.

Veto. The *Veto score* $Sc_V(c)$ of a candidate $c \in A$ is the number of voters that do not rank c last.

Borda. Given an election $(A, \mathcal{R})$ with $|\mathcal{R}| = n$, the *Borda score* $Sc_B(c)$ of a candidate $c \in A$ is given by $Sc_B(c) = \sum_{i=1}^{n} |\{a \in A \mid c \succ_i a\}|$.

***k*-Approval.** For any $k \geq 1$, the *k-Approval score* $Sc_k(c)$ of a candidate $c \in A$ is the number of voters that rank c in the top k positions. Plurality is simply 1-Approval.

Plurality with Runoff. In the first stage, all but two candidates with the top two Plurality scores are eliminated. Then the winner is the one of the survivors that is preferred to the other one by at least half of the voters. We use the *parallel universes* rule (Conitzer, Rognlie, and Xia 2009) for intermediate tie-breaking.

Maximin. Given an election $(A, \mathcal{R})$ with $|\mathcal{R}| = n$, for any $a, c \in A$, let $W(c, a) = |\{i \mid c \succ_i a\}|$. The *Maximin score* $Sc_M(c)$ of a candidate $c \in A$ is given by $Sc_M(c) = \min_{a \in A} W(c, a)$, i.e., it is the number of votes c gets in her worst pairwise election.

Copeland. The *Copeland score* $Sc_C(c)$ of a candidate $c \in A$ is $|\{a \mid W(c, a) > W(a, c)\}| - |\{a \mid W(a, c) > W(c, a)\}|$.

Our Framework

Cloning and independence of clones were previously defined in (Tideman 1987; Zavist and Tideman 1989). However, we need to modify the definition given in these papers in order to model the manipulator's intentions and the budget constraints. We will now describe our setting formally.

Definition 1. *Let $E = (A, (R_1, \dots, R_n))$ be an election with a set of candidates $A = \{c_1, \dots, c_m\}$. We say that an election $E' = (A', (R'_1, \dots, R'_n))$ is obtained from E by replacing a candidate $c_j \in A$ with k* clones *for some $k > 0$ if $A' = A \setminus \{c_j\} \cup \{c_j^{(1)}, \dots, c_j^{(k)}\}$ and for each $i \in [n]$, R'_i is a total order over A' such that for any $a \in A \setminus \{c_j\}$ and any $s \in [k]$ it holds that $c_j^{(s)} \succ'_i a$ if and only if $c_j \succ_i a$.*

We say that an election $E^* = (A^*, \mathcal{R}^*)$ is cloned *from an election $E = (A, \mathcal{R})$ over a set of alternatives $A = \{c_1, \dots, c_m\}$ if there is a vector of non-negative integers $(k_1, \dots, k_m)$ such that E^* is derived from E by replacing each c_j, $j = 1, \dots, m$, with k_j clones.*

Thus, when we clone a candidate c, we replace her with a group of new candidates that are ranked together in all voters' preferences.

The definition above is essentially equivalent to the one given in (Zavist and Tideman 1989); the main difference is that we explicitly model cloning of more than one candidate. However, we still need to introduce the two other components of our model: a definition of what it means for a cloning to be successful, and the budget.

We start with the former assuming throughout this discussion that the voting rule is fixed. To see that this is a non-trivial issue, observe that the final outcome of cloning depends on the relative ranking of the clones chosen by each voter, which is not under the manipulator's control. Thus, a cloning may succeed for some orderings of the clones, but not for others. We can approach this issue from the worst-case perspective, and ask if a given cloning succeeds for *some* ordering, or, more ambitiously, if it succeeds for *all* orderings. Alternatively, we can view our problem from the average-case perspective. That is, we can ask whether one can find a cloning that succeeds with certain probability, assuming that the voters rank the clones randomly and independently, with each ordering of the clones being equally likely. This models the situation where all clones of a given candidate are similar, so that the voters do not really distinguish between them.

Definition 2. *Given a positive real $0 < q \leq 1$, we say that the manipulation by cloning (or simply cloning) is q-*successful *if (a) the manipulator's preferred candidate is not a winner of the original election, and (b) a clone of the manipulator's preferred candidate is a winner of the cloned election with probability at least q.*

The two worst-case approaches discussed above are special cases of this framework. Indeed, a cloning succeeds for all orderings if and only if it is 1-successful, and it succeeds for some ordering if and only if it is q-successful for some $q > 0$; we abuse notation by referring to such cloning as 0-successful. Saying that cloning is 0-successful is equivalent

to saying that the cloning would be successful if the manipulator could dictate each voter how to order the clones. We will use this observation very often as it simplifies proofs.

Observe that, according to our definition, the manipulator succeeds as long as *any one* of the clones of the preferred candidate wins. This assumption is natural if the clones represent the same company (e.g., Coke Light and Coke Zero) or political party. However, if a campaign manager has created a clone of his candidate simply by recruiting an independent candidate to run on a similar platform, he may find the outcome in which this new candidate wins less than optimal. We could instead define success as a victory by the original candidate (i.e., the clone $c^{(1)}$), but in many settings this definition is too restrictive: e.g., if the voting rule is neutral and we are interested in 1-successful cloning, it is essentially equivalent to not allowing to clone the original candidate. Thus, we will use the current definition, but sometimes comment on how our results would change if we used this more stringent definition of victory.

Another issue that we need to address is that of the costs associated with cloning. Indeed, the costs are an important aspect of realistic campaign management, as the manager is always restricted by the budget of the campaign. The most general way to model the cloning costs for an election with the initial set of candidates $A = \{c_1, \dots, c_m\}$ is via a *price function* $p : [m] \times \mathbb{Z}^+ \to \mathbb{Z}^+ \cup \{0, +\infty\}$, where $p(i, j)$ denotes the cost of producing the j-th copy of candidate c_i. Note that $p(i, 1)$ corresponds to not producing additional copies of i, so we require $p(i, 1) = 0$ for all $i \in [m]$. We remark that it is natural to assume that all costs are non-negative (though some of them may equal zero); the assumption that all costs are integer-valued is made for computational reasons.

Definition 3. *An instance of the q-CLONING problem for $q \in [0, 1]$ is given by the initial set of candidates $A = \{c_1, \dots, c_m\}$, a preference profile $\mathcal{R}$ over A, a manipulator's preferred candidate $c \in A$, a parameter $t > 1$, a price function $p : [m] \times [t] \to \mathbb{Z}^+ \cup \{0, +\infty\}$ (with the convention that $p(i, j) = p(i, t)$ for all $i \in [m]$ and all $j > t$), a budget B, and a voting rule $\mathcal{F}$. We ask if there exists a q-successful cloning with respect to $\mathcal{F}$ that cost at most B.*

We require $p(i, j) = p(i, t)$ for $j > t$ to ensure that the price function is succinctly representable. For most voting rules that we consider, it is easy to bound the number of clones needed for 0-successful or 1-successful cloning (if one exists); moreover, this bound is usually polynomial in n and m.

We focus on two natural special cases of q-CLONING:

- ZERO COST (ZC): $p(i, j) = 0$ for all $i \in [m]$, $j \in [t]$ (or, equivalently, $B = +\infty$). In this case we are interested if an election is manipulable at all.
- UNIT COST (UC): $p(i, j) = 1$ for all $i \in [m]$, $j \in \{2, \dots, t\}$. This model assumes that creating each new clone has a fixed, equal cost.

We say that an election E is *q-manipulable by cloning with respect to a voting rule $\mathcal{F}$* if there is a q-successful manipulation by cloning with respect to $\mathcal{F}$ in the ZC model. Further, we say that E is *manipulable by cloning with respect to $\mathcal{F}$* if it is 0-manipulable with respect to $\mathcal{F}$, and *strongly manipulable by cloning with respect to $\mathcal{F}$* if it is 1-manipulable with respect to $\mathcal{F}$.

In the rest of the paper, we discuss the complexity of the q-CLONING problem for a number of well-known voting rules, focusing on the ZC and UC cases. Clearly, hardness results for these special cases also imply hardness results for the general model. Somewhat less obviously, hardness results for the ZC q-CLONING imply hardness results for UC q-CLONING: it suffices to set $B = +\infty$.

For poly-time computable voting rules 0-CLONING is in NP, 1-CLONING is in Σ_2^{p} and q-CLONING is in $\mathrm{NP}^{\mathrm{PP}}$ for $q \in (0, 1)$. However, in this paper we are interested in P-membership and NP-hardness results only.

Plurality and Similar Rules

In this section we focus on q-CLONING for Plurality, Plurality with Runoff, Veto, and Maximin. Surprisingly, these four rules exhibit very similar behavior with respect to cloning.

Plurality We start by considering Plurality which is arguably the simplest voting rule.

Theorem 4. *An election is manipulable with respect to Plurality if and only if the manipulator's preferred candidate c does not win, but is ranked first by at least one voter. Moreover, for Plurality* 0-CLONING *can be solved in linear time.*

Briefly, the optimal algorithm replaces each candidate a with $Sc_P(a) > Sc_P(c)$ by $\left\lceil \frac{Sc_P(a)}{Sc_P(c)} \right\rceil$ clones, and ensures that each clone gets (almost) the same number of votes.

It is not too hard to strengthen Theorem 4 from 0-manipulability to q-manipulability for any $q < 1$. Indeed, if we clone each candidate from $C \setminus \{c\}$ sufficiently many times, then with high probability each clone is ranked first at most $\mathrm{Sc}_P(c)$ times, and c is a winner. However, obviously the manipulator cannot ensure that c always wins: indeed, if all voters order the clones in the same way, the "most popular" clone of each candidate will have the same Plurality score as the original candidate. We summarize these observations in the following proposition.

Proposition 1. *For any $q < 1$, a Plurality election is q-manipulable if and only if the manipulator's preferred candidate c does not win, but is ranked first by at least one voter. However, no election is strongly manipulable.*

Veto and Plurality with Runoff The Veto rule exhibits extreme vulnerability to cloning.

Theorem 5. *Any election in which the manipulator's preferred candidate c does not win is strongly manipulable with respect to Veto. Also, for Veto both* 0-CLONING *and* 1-CLONING *are linear time-solvable.*

Note, however, that if we want the winner to be $c^{(1)}$ (rather that an arbitrary clone of c), then no election is strongly manipulable: whoever vetoed c can still veto $c^{(1)}$.

We now consider Plurality with Runoff. Observe first that cloning an alternative does not change what happens in the runoff: indeed, if c_i beats c_j in the runoff, c_i would beat any

clone of c_j, and if c_i loses to c_j in the runoff, c_i would lose to any clone of c_j. Thus, if an alternative c is a *Condorcet loser*, i.e., for any $a \in A \setminus \{c\}$ a strict majority of voters prefers a to c, then c cannot be made a winner by cloning. Further, c should be able to reach the runoff. Taken together, these two considerations lead to the following criterion.

Theorem 6. *An election is manipulable with respect to Plurality with Runoff if and only if (1) the manipulator's preferred candidate c does not win, and (2) c is not a Condorcet loser and both c and some alternative w that does not beat c in their pairwise election are ranked first by at least one voter. Moreover, for Plurality with Runoff* 0-CLONING *can be solved in polynomial time.*

In fact, we can characterize q-manipulability for $q \in [0, 1]$.

Proposition 2. *For any $q < 1$, an election is q-manipulable with respect to Plurality with Runoff if and only if it is manipulable with respect to it. However, no election is strongly manipulable.*

Maximin Consider an election $E = (A, \mathcal{R})$ with $A = \{a_1, \dots, a_k\}$, $\mathcal{R} = (R_1, \dots, R_k)$, where for $i \in [k]$ the preferences of the i-th voter are given by $a_i \succ_i a_{i+1} \succ_i \dots \succ_i a_k \succ_i a_1 \succ_i \dots \succ_i a_{i-1}$. We will refer to any election that can be obtained from E by renaming the candidates as a *k-cyclic election*. In this election, for any $i = 1, \dots, k$, there are $k - 1$ voters that prefer a_{i-1} to a_i (where we assume $a_0 = a_k$). Thus, the Maximin score of each candidate in A is at most 1. Further, this remains true if we add arbitrary candidates to the election, no matter how the voters rank the additional candidates. This means that by cloning we can reduce the Maximin score of any candidate a to 1: in an election with n voters, we create n clones of a and consider the situation where the voters' preferences over those clones form an n-cyclic election. Thus, a candidate c can be made a Maximin winner as long as she is *Pareto-optimal*, i.e., for any $a \in A$ at least one voter prefers c to a.

Theorem 7. *An election is manipulable by cloning with respect to Maximin if and only if the manipulator's preferred candidate c does not win, but is Pareto-optimal. Further, for Maximin* 0-CLONING *can be solved in linear time.*

It is not clear if one can strengthen the result of Theorem 7 to q-manipulability for $0 < q < 1$. However, it is easy to see that no election is strongly manipulable with respect to Maximin. Indeed, the only way to change a candidate's Maximin score is to clone her. However, after the cloning, all voters may order all clones in the same way, in which case the most popular clone will have the same Maximin score as the original alternative.

Borda, k-Approval, and Copeland

We now consider Borda, k-Approval, and Copeland rules, for which cloning issues get significantly more involved.

Borda Rule For Borda rule, just as for Maximin, Pareto-optimality of the manipulator's favorite alternative is necessary and sufficient for the existence of successful manipulation by cloning.

Theorem 8. *An election is manipulable by cloning with respect to Borda if and only if the manipulator's preferred candidate c does not win, but is Pareto-optimal. Moreover,* UC 0-CLONING *for Borda can be solved in linear time.*

Briefly, an optimal cloning manipulation for Borda in the UC model is to clone c sufficiently many times and ask all voters to order the clones in the same way. However, for $q > 0$, this approach is not necessarily optimal. For example, suppose that c is Pareto-optimal, and, moreover, the original preference profile contains a candidate c' that is ranked right under c by all voters (one can think of this candidate as an "inferior clone" of c; however, we emphasize that it is present in the original profile). Then one can show that by cloning c' sufficiently many times we can make c a winner with probability 1. However, cloning c itself does not have the same effect if the voters order the clones randomly or adversarially to the manipulator. In general, we may need to clone several candidates that are placed between c and its "competitors" in a large number of votes, and determining the right candidates to clone might be difficult. Indeed, it is not clear if a 1-successful manipulation can be found in polynomial time. We thus propose determining the complexity of identifying strongly manipulable profiles with respect to Borda as an open problem.

A related question that is not answered by Theorem 8 is the complexity of 0-CLONING in the general cost model. Note that there is a certain similarity between this problem and that of strong manipulability: in both cases, it may be suboptimal to clone c. Indeed, for general costs, we can prove that q-CLONING is NP-hard for any rational q.

Theorem 9. *For Borda, q-*CLONING *in the general cost model is* NP*-hard for any $q \in [0, 1]$. Moreover, this is the case even if $p(i, j) \in \{0, 1, +\infty\}$ for all $i \in [m], j \in \mathbb{Z}^+$.*

***k*-Approval** Plurality, k-Approval and Borda are perhaps the best-known representatives of a large family of voting rules known as *scoring rules*, i.e., rules in which each voter grants each candidate a certain number of points that depends on that candidate's position in the voter's preference order. (Formally, Plurality, k-Approval, and Borda are *families* of scoring rules.) It would be interesting to characterize scoring rules vulnerable to manipulation by cloning. However, this problem is far from trivial. The next theorem shows that detecting manipulability and strong manipulability is hard for k-Approval for each fixed $k \geq 2$.

Theorem 10. *For each fixed k, $k \geq 2$, for k-Approval it is NP-hard to decide whether an election is manipulable and it is NP-hard to decide whether an election is strongly manipulable.*

Copeland For an election E with a set of candidates A, its *pairwise majority graph* is a directed graph (A, X), where X contains an edge from a to b if more than half of the voters prefer a to b; we say that a *beats* b if $(a, b) \in X$. If exactly half of the voters prefer a to b, we say that a and b are *tied* (this does not mean that their Copeland scores are equal).

For an odd number of voters, the graph (A, X) is a *tournament*, i.e., for each pair $(a, b) \in A^2$, $a \neq b$, we have either $(a, b) \in X$ or $(b, a) \in X$. It turns out that in

this case, we can characterize the set of candidates that can be made winners by cloning in terms of of a well-known tournament solution concept of *uncovered set* (Miller 1977; Fishburn 1977; Laslier 1997), defined as follows. Given a tournament (A, X), a candidate a is said to *cover* another candidate b if a beats b as well as every candidate beaten by b. The *uncovered set* of (A, X) is the set of all candidates not covered by other candidates.

Theorem 11. *For any $q \in [0, 1]$, an election E with an odd number of voters is q-manipulable with respect to Copeland if and only if the manipulator's preferred candidate c does not win, but is in the uncovered set of the pairwise majority graph of E.*

Proof. Consider an election E with the set of alternatives A, $|A| = m$, and an odd number of voters. Let (A, X) be its pairwise majority graph.

Suppose that c is covered by some $a \in A$, and so $Sc_C(c) < Sc_C(a)$. Creating k clones of an alternative x increases by $k-1$ the score of each alternative that beats x and decreases by $k-1$ the score of the alternatives that are beaten by x. Moreover, if we replace x with k clones, the scores of all clones of x will be at most $Sc_C(x)+k-1$. Thus, no matter which alternatives we clone, we cannot close the gap between a and c (or its highest-scoring clone).

Conversely, suppose that c is in the uncovered set. Let $U(c)$ denote the set of all alternatives that beat c, and let $D(c)$ denote the set of all alternatives that are beaten by c. First, we create $2m+1$ clones of c. This lowers the score of each alternative in $D(c)$ by $2m$ and raises the score of each alternative in $U(c)$ by $2m$. On the other hand, by a simple counting argument, there exists some clone of c whose score is now greater or equal to the original score of c; denote this clone by c'. Let $Sc'_C(x)$ denote the score of an alternative x at this stage; we have $Sc'_C(c') > Sc'_C(x)$ for any $x \in D(c)$, $Sc'_C(c') > Sc'_C(x) - (4m-2)$ for any $x \in U(c)$.

Now create $4m+1$ clones of each alternative in $D(c)$. This increases c''s score by $4m|D(c)|$. Further, for any $a \in D(c)$, the score of any clone of a constructed at this stage exceeds that of $Sc'_C(a)$ by at most $4m|D(c)|$. Thus, at this stage, c' has a higher Copeland score than any of the newly-generated clones. Finally, since c was not covered, no candidate in $U(c)$ beats all candidates in $D(c)$. Thus, the last step increased the scores of all candidates in $U(c)$ by at most $4m(|D(c)|-1)$. It follows that c' now has a higher Copeland score than any candidate that is not a clone of c. □

For elections with an even number of voters, the situation is significantly more complicated. In more detail, the notion of uncovered set can be extended to pairwise majority graphs of arbitrary elections in a natural way (see, e.g. (Brandt and Fischer 2007)). However, for an even number of voters, the condition that c is in the uncovered set turns out to be necessary, but not sufficient for manipulability by cloning.

On the other hand, finding an optimal-cost cloning manipulation is hard even in the UC model.

Theorem 12. *For Copeland,* UC q-CLONING *is* NP*-hard for each $q \in [0, 1]$.*

Detecting Clones

So far we have considered the problem of introducing clones so that a given candidate becomes a winner. In this section we focus on the reverse issue: Given an election, we would like to know if it could have been obtained from a smaller one via cloning. This is important from the point of view of a candidate who wants to defend herself from a manipulator performing cloning: Before she can apply any countermeasures, she has to determine whether clones exist.

Theorem 13. *Given an election $(A, \mathcal{R})$ and an integer $m \leq |A|$, it is possible to detect in polynomial time if there is an election $(A', \mathcal{R})$ such that $|A'| = m$, $A' \subseteq A$, and $(A, \mathcal{R})$ could have been cloned from $(A', \mathcal{R})$.*

Note that, in effect, the above theorem speaks of finding a set cover of A with exactly m disjoint sets of possible clones. Such covering problems are typically NP-hard, but in our case we use the fact that: (a) in each election with $|A|$ candidates there are at most $O(|A|^2)$ easily computable sets of candidates that can possibly be clones of each other, and (b) each voter ranks clones in consecutive blocks, so we can use any voter's preference order to guide the search for the cover. Our algorithm is based on dynamic programming and, in fact, it can solve much more intricate problems than given in the theorem statement. For example, we can exclude some groups of candidates from consideration, or we can attach a weight to each set of possible clones and ask for a cover that maximizes the total weight of possible clones used (e.g., the weights can model our beliefs on how likely it is that candidates in a given group are indeed clones).

Theorem 13 allows us to detect elections where clones could have been introduced. However, we may also want to know if our favorite candidate could have been a winner prior to the introduction of the clones.

Definition 14. *Let $\mathcal{F}$ be a voting rule. In the* DELETING-CLONES *problem for $\mathcal{F}$ we are given an election $E = (A, \mathcal{R})$, a designated candidate $c \in A$, and an integer t, and we ask if there is a set $A' \subseteq A$ such that $c \in A'$, $|A'| \geq |A| - t$, c is an $\mathcal{F}$-winner of $E' = (A', \mathcal{R})$ and E could have been cloned from E'.*

A "yes"-instance of the DELETING-CLONES problem corresponds to a situation where it is possible that someone stole the election from c by introducing at most t clones. Clearly, c's campaign managers would like to be able to detect this so as to remove the clones from the election[1]. Interestingly, DELETING-CLONES is easy for some of those rules for which 0-CLONING is easy. We conjecture that DELETING-CLONES is NP-complete for Copeland, Borda and k-approval, for each fixed $k > 1$.

Theorem 15. DELETING-CLONES *is in* P *for Plurality, Maximin, and Veto.*

Cloning Versus Adding Candidates

We conclude this paper by comparing cloning to the well-studied problem of control via adding candidates.

[1] For example, if we detect tampering with the election then the clones may be forced to withdraw by the center, or we could try to discredit them in the eyes of the voters.

Voting rule	AC control	0-Cloning	1-Cloning
Plurality	NPC	P	—
Maximin	NPC	P	—
Plurality w/Runoff	NPC	P	—
Veto	NPC	P	P
Borda	NPC	P	?
k-Approval, $k \geq 2$	NPC	NPC	NP-hard
Copeland	NPC	NPC	NP-hard

Table 1: The complexity of control via adding candidates and of q-CLONING in the UC model for $q \in \{0, 1\}$. Note that for Plurality, Plurality with Runoff and Maximin 1-successful cloning is impossible. The complexity of AC control has been established for Plurality in (Bartholdi, Tovey, and Trick 1992), for Maximin in (Faliszewski, Hemaspaandra, and Hemaspaandra 2009), and for Copeland in (Faliszewski et al. 2009). All cloning results and most of the AC control results hold in both unique-winner and non-unique winner models.

Definition 16. *Let $\mathcal{F}$ be a voting rule. In the* constructive control by adding candidates *problem we are given an election $(C \cup A, V)$, where $A \cap C = \emptyset$, a designated candidate $c \in C$, and a nonnegative integer k. We ask if there is a set $A' \subseteq A$, such that $|A| \leq k$, and c is the unique winner of election $(C \cup A', V)$.*[2]

In the definition above, the set C corresponds to already registered candidates and the set A is the set of "spoiler" candidates that the manipulator can introduce into the election.

While q-CLONING and AC control are similar in that both of them deal with adding new candidates, neither of these problems is a special case of the other. Indeed, they place different restrictions on the candidates to be added and their positions in the votes. Specifically, in q-CLONING the new candidates must be clones of an existing candidate, but (especially in 0-CLONING) we have some freedom as to how to arrange the new candidates in the votes. In contrast, in AC control problems, the spoiler candidates need not be adjacent to each other in all votes, but the order of all the candidates in each vote is prespecified.

Intuitively, cloning problems appear to be somewhat more natural, as well as computationally easier than the corresponding AC control problems. Indeed, we can construct contrived instances of AC control by placing the spoiler candidates in any way we like, so as to facilitate computational hardness proofs. This intuition is verified—at least to some degree—in Table 1, where we compare the complexity of AC control and of q-CLONING in the UC model. (The UC model is, in essence, the exact analog of counting the number of candidates added in the AC control problems.)

To compare q-CLONING and AC control, in addition to the results from the literature, we provide the next theorem.

Theorem 17. *The constructive control by adding candidates problem is* NP*-complete for Borda, Veto, Plurality with runoff and, for each fixed k, for k-Approval.*

See (Russel 2007) for some other results on control in Borda.

Our work is also related to that of Maudet et al. (2010) who ask if a given candidate can become a winner after some new candidates are added. In contrast to our work, they do not require the new candidates to be clones of the existing ones; however, unlike in control by adding candidates, they allow new candidates to be placed arbitrarily in the votes.

[2]Unique-winner model is standard for control problems.

Conclusions

We have provided a formal model of manipulating elections by cloning, characterized manipulable and strongly manipulable profiles for many well-known voting rules, and explored the complexity of finding a minimum-cost cloning manipulation. The grouping of voting rules according to their susceptibility to manipulation by cloning differs from most standard classifications of voting rules: e.g., scoring rules behave very differently from each other, and Maximin is more similar to Plurality than to Copeland. Future research directions include designing approximation algorithms for the minimum-cost cloning under voting rules for which this problem is known to be NP-hard, and extending our results to other voting rules.

References

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

Brandt, F., and Fischer, F. 2007. Computational aspects of covering in dominance graphs. In *Proc. of AAAI-07*.

Conitzer, V.; Rognlie, M.; and Xia, L. 2009. Preference functions that score rankings and maximum likelihood estimation. In *Proc. of IJCAI-09*.

Faliszewski, P.; Hemaspaandra, E.; Hemaspaandra, L.; and Rothe, J. 2009. Llull and Copeland voting computationally resist bribery and constructive control. *JAIR* 35:275–341.

Faliszewski, P.; Hemaspaandra, E.; and Hemaspaandra, L. 2009. Multimode attacks on elections. In *Proc. of IJCAI-09*.

Fishburn, P. 1977. Condorcet social choice functions. *SIAM Journal on Applied Mathematics* 33(3):469–489.

Laffond, G.; Laine, J.; and Laslier, J. 1996. Composition consistent tournament solutions and social choice functions. *Social Choice and Welfare* 13(1):75–93.

Laslier, J.-F. 1997. *Tournament Solutions and Majority Voting*. Springer-Verlag.

Maudet, N.; Lang, J.; Chevaleyre, Y.; and Monnot, J. 2010. Possible winners when new candidates are added: The case of scoring rules. In *Proc. of AAAI-10*. To appear.

Miller, N. 1977. Graph theoretical approaches to the theory of voting. *Am J Polit Sci* 21(4):769–803.

Russel, N. 2007. Complexity of control of Borda count elections. Master's thesis, Rochester Institute of Technology.

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

Tideman, T. 1987. Independence of clones as a criterion for voting rules. *Social Choice and Welfare* 4(3):185–206.

Zavist, T., and Tideman, T. 1989. Complete independence of clones in the ranked pairs rule. *Social Choice and Welfare* 64(2):167–173.