Adapting the Social Network to Affect Elections

Sigal Sina
Dept of Computer Science
Bar Ilan University, Israel
sinasi@macs.biu.ac.il

Noam Hazon
Dept of Computer Science
and Mathematics
Ariel University, Israel
noamh@ariel.ac.il

Avinatan Hassidim
Dept of Computer Science
Bar Ilan University, Israel
avinatanh@gmail.com

Sarit Kraus
Dept of Computer Science
Bar Ilan University, Israel
sarit@cs.biu.ac.il

ABSTRACT

We investigate the effect a social network could have on voting outcomes. We consider a group of self-interested agents where each agent has a strict preference order over a set of outcomes. Each agent votes strategically, taking into consideration both her preferences, and her (limited) information about the preferences of other voters. We assume that the information the agent has comes from her friends in the social network and from a public opinion poll. If agents were not strategic at all, the social network (and the poll) would not matter, since they would just vote according to their preferences. However, if the agents deviate and vote strategically the network plays a great effect. To measure this effect, we focus on iterative voting with Plurality voting rule. We show, both in theory and in simulations, that for many networks, adding a linear number of edges can make any outcome the winner. We view our results as yet another indication to the effect that a central organizer, such as a company who controls social media, could have on our lives - by introducing us to certain people it can affect our information and our decisions.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multi-agent Systems

Keywords

Social Choice; Iterative Voting; Social Networks

1. INTRODUCTION

Voting systems have been used by people for centuries as tools for group decision-making in settings as diverse as politics [17] and entertainment [6]. More recently, computers have used voting and rank aggregation methods for applied tasks such as aggregating search results from the web [4]. In both cases, it is common that voters will exchange opinions before they take the decision on how they should vote. It is thus natural to model voting as a dynamic iterative process, where voters discuss and share information among themselves possibly over the course of several rounds, rather than as a one-shot game.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

In recent years, social networks have surged in popularity. Social networks enable people to share information easily and interact with each other. It is thus crucial to understand the effect social networks could have on voting outcomes. Another very important source of information is public polls. Polls are very common in political environments, as they provide a representative sample of the electorate and can thus cause some voters to change their vote.

As an example, consider the first state in the nominations race for the Democratic (Republican) party. In this setting, there are several contenders, and a population of voters (members of the party) who need to choose their candidates. Before election day, there are usually polls, but voters also talk among themselves or via social networks, to know the status. Finally, if a voter thinks that two candidates have a high chance of winning, she may vote for one of them and not for her true favorite candidate (indeed we witness candidates leaving the race at some point). Note that the population of voters is much larger than the size of the poll.

In this paper we study the effect a social network could have on voting outcomes, and whether a central organizer can utilize this effect to its advantage. To do this, we consider a simple model, in which there are n voters who wish to choose one of m candidates, using the Plurality rule. Each voter has her own preferences, and the voters are connected in a social network. In addition, there is a poll, which is public information. We use the model of iterative voting, which proceeds by rounds. At the beginning of every round, each voter learns what her neighbors planned to vote in the previous round. Then she either keeps her previous planned vote, or changes it. A voter changes her vote, only if she believes that she can affect the election outcomes, after taking into account both the votes of her friends and the poll. At some later point in time, a ballot is cast, and the winner is decided. Note that the network structure has no effect on the preference of the voters, who are self-interested.

To understand the power of the social network, we show that under very general conditions, small changes in the network, can lead to dramatic changes in the identity of the winner. To do this, we show a polynomial time algorithm which takes a candidate ω and makes it the winner, by only adding a linear number of edges. Our algorithm only requires that:

- The fraction of the population who do not consider ω to be the last on their preference list is greater than $1/m$.
- Each candidate has a minimal number of voters who support it, where this number is independent of the size of the network.

Our algorithm still succeeds in making ω the winner even if the poll is adversarial, i.e., it is constructed in such a way that makes it harder for ω to become a winner. Moreover, the poll can be chosen

in an adversarial manner after the algorithm acts, and still ω would win (we will then need an upper bound on the size of the poll and a sub-linear number of voters that support every candidate). In addition, the algorithm does not know exactly when the ballot will be cast. To do this, it succeeds in a harder task, of stabilizing the network such that no voter would like to change her vote, given the local information she has. Finally, the initial network is almost adversarial. We only require that most nodes have a bounded degree, and that for each candidate c, there is no single voter who is connected to most of the supporters of c. In particular, if no voter is connected to ϵn of the population, and every candidate has $O(n)$ supporters, the latter condition holds.

The consequence that even small changes in the network can lead to dramatic changes in the election outcome leads to an inevitable result. Using the proposed algorithm, a central election organizer that has access to the preferences of all the voters can easily (in polynomial time) adapt the network to its advantage. However, such an organizer will still be restricted by the number of edges that it is able to add to the social network. We thus present a greedy heuristic, which is based on our algorithm, that aims at making a desirable candidate ω the winner while minimizing the number of edges that are added to the network, and the number of voters who are involved in this adaption of the network. The obvious trade-off is that there is no guarantee that the heuristic will find the desired way to adapt the network in some instances, even if it is possible. However, using extensive simulations on generated and real profiles and with real networks, we found that our heuristic always succeeds in finding a relatively small number of edges to add to the network that makes a candidate ω the winner.

We would like to emphasize that in this paper we do not assume that the voter is similar to her neighbors, or that her true opinion is dependent on the opinions or actions of her neighbors. Indeed, since the social network affects the votes by simply spreading information, our results apply to any communication network that dissipates information between agents. In particular, taking the "Voting on a Meeting Time" example [5], in which agents use an iterative voting procedure to agree on a meeting time, these results imply that the communication network could have a profound effect on the outcome.

2. RELATED WORK

We study elections under the framework of *iterative voting* that has been the subject of numerous studies in recent years. Meir et al. [15] studied iterative voting for the Plurality rule, and additional rules were studied by Lev et al. [12]. Since they show that the iterative version of many voting rules does not converge, the works of [9] and [16] considered restrictions on the way the voters are allowed to update their votes. Further work studies the strategies actually used by people in an iterative voting process [20]. All of these works assume that voters receive at least some information regarding the true preferences and the votes of all the other voters in every round, while we assume that the voters belong to a social network and are able to see only the votes of their friends. A recent work [14] assumed that voters receive even less than that. Namely, they only have an estimated, uncertain view of the electorate. They thus proposed a simple behavioral heuristic, which guarantees convergence of the iterative process. A model of knowledge very similar to ours was proposed by Chopra et al. [3] (but with directed graph). However, they neither define how the voters adapt their votes given their knowledge, nor do they analyze any computational consideration.

There are several studies that analyze the influence of friends in social networks in the framework of iterative voting. In the model presented in [10] the voters are cooperative and try to maximize the group's overall benefits. In [8], [1] and in [13] the sincere preferences of the voters are influenced by the votes of their friends, i.e., it is possible that a voter will have such a utility function that she will prefer to vote for her least preferred candidate, to comply with the votes of her friends. Their motivation is that people care about their "public image" as determined by their votes and the socially accepted outcome (the chosen winner). In our work, voters are self-interested, and they thus may change their voting strategy, but not their own true preferences. Therefore, in our model a voter will never vote for her least preferred candidate.

One consequence of our analysis is that a central election organizer that has access to the preferences of all the voters can easily adapt the network to its advantage. The work of [21] proposed an epistemic knowledge based framework for strategic voting, and discussed the modeling of a central authority that can try to change the election outcome by revealing information. In a different domain, the work of [19] considers the setting of a central organizer that can add edges to a network in order to affect the outcome of a coalitional game. We note that even though it is commonly assumed that an election manipulator has full information on the electorate (see [23], [22] and [11] for example), this assumption is even more reasonable in our setting, since current social networks try to learn the preferences of their users. For example, Facebook succeeded in adding new edges to the network with their friends suggestions that are based on the users preferences [18, 2].

3. PRELIMINARIES AND DEFINITIONS

We have a set of *candidates* (also referred to as *alternatives*) $C=\{c_1,\ldots,c_m\}$ of size m and a set of *voters* $V=\{v_1,\ldots,v_n\}$ of size n. Each voter v has a *preference order* (i.e., a ranking) over C, which we denote $\succ_v$. We state that voter v *dislikes* candidate c if v ranks c last in $\succ_v$. At an election, each voter submits a preference order bl_v, which does not necessarily coincide with $\succ_v$. We refer to bl_v as the vote or ballot of voter v. The vector $\mathcal{B}=(bl_1,\ldots,bl_n)$ is called a *preference profile*. A *voting rule* $\mathcal{F}$ is a mapping from the set of all preference profiles to the set of candidates. In this paper, the voting rule $\mathcal{F}$ is taken to be the Plurality rule, where each voter assigns a score of one to the candidate that is ranked first in bl_v and a score of zero to all the other candidates. We will also assert that voter v *votes* for candidate c if v ranks c first in bl_v. The score of candidate $c\in C$ under Plurality is denoted s^c_{truth}, and it is simply the number of voters that vote for c. The winner of the election is selected from the candidates with the highest score using a tie-breaking rule. To simplify the analysis, we assume that the tie-breaking rule is *lexicographic*, i.e., given a set of tied candidates, it selects one that is maximal with respect to a fixed ordering.

We study elections under the framework of *iterative voting*, which proceeds by rounds. As in regular elections, the voters express their true preferences in the first round. In the next round all the voters are allowed to change their votes (simultaneously) if they believe that by doing so the result will change in their favor (we will formally define it latter). The process repeats until it eventually reaches a convergence state, i.e., a profile where no single voter believes that she can get a better result by changing her vote, and then the ballot is cast and a winner is announced. Unlike previous work, we do not assume that the voters receive any global information from the election organizer on the votes of the electorate. Instead, each voter is acquainted with the votes of her friends and with the result of an initial opinion poll.

Let $G=(V,E)$ be an undirected graph (without self-loops) representing the relations between voters. Each voter $v\in V$ is familiar with the voters in her neighborhood, $\mathcal{N}_v=\{u|(v,u)\in E\}$, and thus it is assumed that v will know their votes at the end of each round of

the election process. Let $n_v^c(r)$ be the number of neighbors of voter v in G that vote for candidate c in round r. We denote the degree of v in G, that is $|\mathcal{N}_v|$, by d_v. In our algorithm we will use a bound, denoted by d, on the degree of the voters. We will thus say that v is a *bounded degree* voter if $d_v \leq d$. Otherwise, v is an *unbounded degree* voter.

In addition, there is a poll that is conducted before the election starts. Let s_{poll}^c be the score candidate c receives in the poll, let $max\Delta$ be the difference between the maximum and minimum scores of the candidates in the poll, and let $s=\sum_{c\in C} s_{poll}^c$ be the size of the poll. We assume that after the first round of the election all the voters are informed of the poll results, which together with the votes of their friends composes the voters point of view on the possible election outcome. We use $s_v^c(r)$ to denote the score of candidate c according to voter v's perspective in round r, which is $s_v^c(r)=s_{poll}^c+n_v^c(r)$. Therefore, at each round, each voter v computes its $s_v^c(r)$ for each candidate c and then the identity of the winner, denoted by c_v^{win}. If there is a candidate ω such that $\omega \succ_v c_v^{win}$ and v is able to change her vote in any way such that ω will win in the next round, then v will do so (if there is more than one such candidate then v will select the most preferred candidate). We state that voter v *supports* candidate c from round r, if there is an index $r>0$, such that for every round $r' \geq r$, v votes for c. If a voter supports a candidate from the first round, this voter is considered a *supporter*.

We allow our algorithm to make an acquaintance between any two of the voters, which will create a new link between the two voters in the social networks. However, the links are added only once and before the election process starts. Since all the voters are assumed to vote according to their true preferences in round 1, the added edges may affect the voters only from round 2 and on. A pictorial representation of the iterative process and of when information becomes available is depicted in Figure 1.

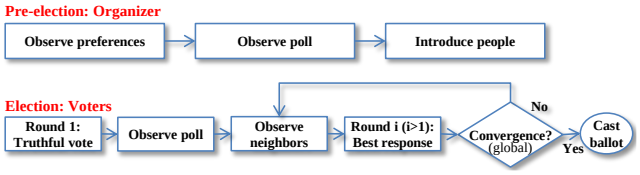


Figure 1: The iterative process flow chart

Let $G'=(V,E')$ be the new social network after the addition of the edges to G. To simplify notations, hereafter we use $n_v^c(r)$ to denote the number of neighbors of voter v in G' (and not in G) that vote for candidate c in round r, and $n_v^c(0)$ denotes the number of neighbors of voter v in G, which rank c first in their true preference order. We are now ready to define our basic problem.

DEFINITION 1. *In the* INFLUENCE *problem we are given a set C of candidates, a set V of voters specified via their preferences, a voting rule $\mathcal{F}$, a social network G, poll results for each $c\in C$, and one specific candidate ω. We are asked to find a set of edges that will be added to G, such that ω will be the winner of the iterative process.*

We will also consider the case in which the poll is given to the voters but not to the algorithm or central organizer, and we call this setting an *unknown poll*.

4. KNOWN POLL

We start by establishing two lemmas, which serve as basic building blocks for our algorithm. In the first lemma we show how (and when) it is possible to make a voter vote for a specific candidate. The idea is quite intuitive: if we would like to cause a voter v to vote for candidate ω and v dislikes candidate a, we can connect v to many supporters of a and ω that will (falsely) make v believe that a will win unless v will change her vote to ω.

LEMMA 1. *Given a desired candidate ω and a voter v such that v does not vote for ω in round 1, it is possible to influence v such that v will support ω from round 2 if:*

1. *v dislikes $a\in C$, $a\neq\omega$.*
2. *For each candidate $\ell\in\{a,\omega\}$ there are at least k supporters of ℓ, $k=2(max\Delta+d_v+1)$.*

PROOF. Let α be 0 if $a\succ\omega$ in the *lexicographic* tie-breaking rule and 1 otherwise, and let $\delta^{a\omega}$ be the value of $(s_{poll}^a+n_v^a(0)-(s_{poll}^\omega+n_v^\omega(0))-\alpha)$. If $\delta^{a\omega}>0$, the organizer connects v to $\delta^{a\omega}$ supporters of ω. Otherwise, the organizer connects v to $|\delta^{a\omega}|$ supporters of a. In addition, the organizer connects v to $max\Delta+d_v+1$ supporters of a and $max\Delta+d_v+1$ supporters of ω. Thus, v's perspective after round 1 is that a will win the election, if ω gets one more vote it will win the election, and both a and ω have a score that is greater than the score of any other candidate. Since $\omega\succ_v a$, v will vote for ω in round 2. In addition, even if all the voters from $\mathcal{N}_v$ change their vote in the next rounds the scores of any candidate $c\in C\setminus\{a,\omega\}$ will not reach the scores of a or ω according to v's belief. Thus, v will not change her vote from round 2 and on. We required that $k=2(max\Delta+d_v+1)$ to ensure that there are sufficient supporters of a and ω. □

We note that some of the edges that we would like to add to v may already be in the original network G. Our construction still works, and we only need to treat these edges as ones that belong to G' and update the $n_v^c(0)$ values for the relevant candidates $c\in C$. This principle is true for all our proofs, unless we explicitly state otherwise.

The next lemma claims that we can influence a voter to keep her current vote, without adding too many edges. This will be useful later, to ensure that a voter who has a very high degree in G, i.e., an unbounded degree voter, will not change her vote. The idea of our proof is to ensure that the influenced voter will not see any tie between any two candidates, in any round. However, this is not trivial, since breaking a tie between two candidates in a given round might create a new tie between two other candidates in a different round.

LEMMA 2. *Given a voter v, it is possible to influence v so that v will be a supporter if:*

1. *For every $u\in\mathcal{N}_v$, u changes her vote in at most R rounds.*
2. *For every $u\in\mathcal{N}_v$, we are familiar with the votes of u until the round where u starts to support a specific candidate.*
3. *For each candidate ℓ there are at least k supporters of ℓ, $k=3(m-1)(R+1)$, that are not connected to v in G.*

PROOF. We use S^c to denote a set of supporters of candidate c. Now, consider Algorithm 1. The algorithm first computes the difference in the scores of the first two candidates c_1 and c_2 (according to v's perspective), denoted by δ, in the first round and in all the R rounds (Lines 5-9). If at least one of the δ's is zero, one or minus one (i.e., there is at least one round where a tie is possible), there is a need to shift all the δ's by one and check again. This shift corresponds to adding an edge between S^{c_2} and v that will increase the score of c_2 by one. This process repeats until the algorithm finds a shift where there is no possibility for any tie (Lines 10-14). Since there are at most $R+1$ different δ's, according to pigeonhole principle the algorithm is guaranteed to find such a shift in at most $3(R+1)$ steps. The first iteration ends when the algorithm adds the links from S^{c_2} and updates the scores accordingly (Lines 15-16). In the next iteration the algorithm computes the difference in the scores of c_1 and c_3 and the difference in the scores of c_2 and c_3. There are at most $2(R+1)$ such numbers, thus the algorithm is

guaranteed to find a suitable shift, adding at most $6(R+1)$ edges, to increase the score of c_3 so that there will not be any tie. The next iteration handles the differences in the scores between the first three candidates and the forth candidate, and so on. Overall, the maximum shift is $3(m-1)(R+1)$ and thus the algorithm needs at most this number of supporters for each candidate. □

Algorithm 1 Make-Supporter

Require: a specific voter, $v \in V$
1: for each $\ell \in C$, $S^\ell \leftarrow$ a set of k supporters of ℓ that are not connected to v
2: $\mathcal{R} \leftarrow$ the indexes of the R rounds
3: insert 1 into $\mathcal{R}$
4: **for** $i \leftarrow 2$ **to** m **do**
5: $\quad\Delta \leftarrow \emptyset$
6: $\quad$**for** $j \leftarrow 1$ **to** $i-1$ **do**
7: $\quad\quad$**for all** $r \in \mathcal{R}$ **do**
8: $\quad\quad\quad\delta \leftarrow s_v^{c_i}(r) - s_v^{c_j}(r)$
9: $\quad\quad\quad$insert δ to Δ
10: $\quad sh \leftarrow 0$
11: $\quad$**while** $\exists\delta \in \Delta, |\delta| < 2$ **do**
12: $\quad\quad sh \leftarrow sh + 1$
13: $\quad\quad$**for all** $\delta \in \Delta$ **do**
14: $\quad\quad\quad\delta \leftarrow \delta + 1$
15: $\quad$connect sh voters from S^{c_i} to v
16: $\quad$update $s_v^{c_i}(r)$ for all $r \in \mathcal{R}$

We are now ready to solve the Influence problem. Our algorithm (Algorithm 2) handles several types of voters. The first set of voters includes those that we ensure will be supporters (by creating a clique among them). We use the supporters to affect the rest of the voters, whom we call *affected voters*. Let n' be the number of affected voters. We divide the affected voters into four groups. The first group of voters consists of bounded degree voters who vote for ω in the first round, and we call them type I voters. We simply connect them to a sufficient number of supporters who make them keep their votes. The second and third groups consist of bounded degree voters whose most preferred candidate is not ω. In one group, which we call type II voters, there are voters who do not dislike ω, and we thus use Lemma 1 to make them support ω from round 2. In the other group, which we call type III voters, there are voters who dislike ω, and we thus use Lemma 1 to distribute their votes as uniform as possible among the $(m-1)$ candidates other than ω. In the last group of voters, which we call type IV voters, there are unbounded degree voters. We use Lemma 2 to ensure that they will not change their votes, and we will thus be able to reach a stable state. Overall, our algorithm guarantees that the election will converge in the third round, and the winner will be ω. That is:

THEOREM 3. *Let d be a natural number, $z=max(6(m-1),2(max\Delta+d+1))$, and $n'=n-zm^3$.* INFLUENCE *can be solved in polynomial time if:*

1. *There is at least a fraction p of affected voters with a degree of at most d, where $1-\frac{2max\Delta}{n'} < p \leq 1$.*
2. *There is at least a fraction p_ω of affected voters with a degree of at most d that do not dislike ω, where $p_\omega > max(\frac{1}{m}, (1-p))$.*
3. *For each candidate ℓ there are at least zm^2 voters with a degree of at most d, who vote for ℓ in the first round. In addition, at least a fraction $\frac{1}{m}$ of these voters do not have any voter v in their neighborhood with $d_v > d$.*

PROOF. We use Algorithm 2, which guarantees that the election converges in the third round, and the winner is ω. To prove correctness we first need to show that no voter has an incentive to change her vote after round 2. Assume that all the voters from S are supporters. Then, for every voter v that is a type I voter, $s_v^\omega(r) > s_v^c(r)$ for every candidate $c \in C \setminus \{\omega\}$ and for every round r, since v is connected to $z/2$ supporters of ω. That is, v believes that ω will win the election no matter how the voters in her neighborhood shall vote, and thus v will not have an incentive to change her vote. Every voter v that is a type II or type III voter will change her vote in round 2 to ω or a, respectively, according to Lemma 1, which we can use since all the voters from S are supporters. Lemma 1 also guarantees that v will not change her vote in the next rounds. In addition, every voter v that is a type IV voter will be a supporter according to Lemma 2, which we can use since if v votes for c, all the voters from $S^\ell_{(c,c)}$ for all $\ell \in C$ are supporters that are not connected to v, and any other voter in G will change her vote in at most one round (i.e., type II or type III voters, in round 2). We now show that all the voters from S are indeed supporters.

Note that each voter $v \in S^\omega_{(\omega,\omega)}$ is connected to at least z voters who will vote for ω, and to at most d other voters. Therefore, v will not have an incentive to change her vote. Each voter $v \in S^\ell_{(i,j)}, i \neq j$, is connected to at least z voters who will vote for ℓ and to possibly many voters who will vote for i in round 1 and for j in round 2. Since the sets $S^\ell_{(i,j)}$ are pairwise disjoint, v is not connected to any other voter except to at most d voters from $\mathcal{N}_v$, which effectively do not have any impact on v's vote. Therefore, after round 1 there is no candidate $c \in C \setminus \{\ell, i\}$ that will reach the score of ℓ or i according to v's belief. Even if v believes that there is a tie between ℓ and i, v cannot affect the election outcome and thus v will not have an incentive to change her vote after round 1. Using a similar argument for round 2 and candidates ℓ and j we conclude that v will not have an incentive to change her vote in any round. Finally, each voter $v \in S^\ell_{(i,i)}$ is connected to z voters that vote for ℓ and to at most $2max\Delta$ type IV voters that do not change their votes. Therefore, v will not have an incentive to change her vote. Thus every $v \in S$ is a supporter.

It remains to show that ω will win the election after round 2. The set of supporters S gives the same number of points for each candidate, and we thus need to consider only the set of affected voters. We required that there is at least a fraction p_ω of type I and II voters, and Algorithm 2 ensures that they support ω from round 2. There is at most a fraction $1-p$ of type IV voters that are supporters, and they may vote for candidates other than ω. In addition, there is at most a fraction $p-p_\omega$ of type III voters, and Algorithm 2 distributes their votes as uniform as possible among the $m-1$ candidates other than ω, given the votes already casted by the type IV voters. Clearly, each candidate $c \in C \setminus \{\omega\}$ will get at most a fraction $\frac{(p-p_\omega)+(1-p)}{m-1}$ of votes, if there are enough votes to distribute, or $(1-p)$ otherwise. Therefore, if $p_\omega > \frac{(p-p_\omega)+(1-p)}{m-1}$ and $p_\omega > (1-p)$, that is $p_\omega > max(\frac{1}{m}, (1-p))$ as we required, then ω will win the election. □

We now discuss some lower bounds for the effect of the network, explaining why the conditions required by the algorithm are almost tight. We required that at most $1/m$ of the voters dislike ω. Indeed, since no voter can ever be made to vote for a candidate she dislikes (no matter what her friends do), if the fraction of voters who do not dislike ω is less than $1/m$, then ω will never win. Now, consider the case in which everybody's first preference is a, and everyone's second preference is ω. No algorithm can convince any voter to vote for ω, since starting from the first round, every voter would only see other voters voting for a. Hence, we require that every candidate has some supporters. In this work we assume that the number of voters go to infinity, but that most nodes have a constant degree, and that the number of candidates and size of the poll are constant. Hence, we only tried to show that we need that each candidate has a constant number of supporters, without actually trying to optimize the constant. We leave this refinement for future work.

Algorithm 2 Influence (known poll)

for all $\ell, i, j \in C$ **do**
  $S^{\ell}_{(i,j)} \leftarrow$ a set of z voters who vote for ℓ in round 1, where $\forall v \in S^{\ell}_{(i,j)}, d_v \leq d$
  create a clique between the voters in $S^{\ell}_{(i,j)}$
ensure that the sets $S^{\ell}_{(i,j)}$ are pairwise disjoint
define $S =$ all the voters in the sets $S^{\ell}_{(i,j)}$
for all $c \in C$, initialize $s^c_{curr} \leftarrow s^c_{truth}$
for all $v \in V \setminus S$, where $d_v \leq d$ **do**
  if v vote for ω in round 1 **then**
    connect v to $z/2$ voters from $S^{\omega}_{(\omega,\omega)}$ {*type I voters*}
  else
    define $c =$ the candidate that v votes for in round 1
    $s^c_{curr} \leftarrow s^c_{curr} - 1$
    if v does not dislike ω **then**
      define $a =$ the candidate that v dislikes
      make v vote for ω in round 2, using $S^{\omega}_{(c,\omega)}$ and $S^{a}_{(c,\omega)}$ and the construction of Lemma 1 {*type II voters*}
    else
      define $a = argmin_{\ell \in C \setminus \{\omega\}}(s^{\ell}_{curr})$
      make v vote for a in round 2, using $S^{\omega}_{(c,a)}$ and $S^{a}_{(c,a)}$ and the construction of Lemma 1 {*type III voters*}
      $s^a_{curr} \leftarrow s^a_{curr} + 1$
for all $v \in V$, where $d_v > d$ **do**
  define $c =$ the candidate that v votes for in round 1
  make v a supporter, using $S^{\ell}_{(c,c)}$ for all $\ell \in C$ that are not connected to v, and Lemma 2 {*type IV voters*}

How many edges should the algorithm add? If the difference between the current winner a and ω is k, then clearly the algorithm needs to convince at least $k/2$ voters to change their vote, and hence it must add at least $\Omega(k)$ edges. We show that the algorithm adds $O(k)$ edges, where again the constant could depend on degree bounds, on the number of alternatives and on the size of the poll. Again we leave the problem of optimizing the exact number of edges for future work.

5. UNKNOWN POLL

To us, the most surprising result is that the network effect is so strong, that any candidate ω can be made to win regardless of the results of the poll. Consider some voter v, whose most preferred candidate is a, and whose least preferred candidate is b. Suppose that $b \succ \omega$ by the tie-breaking rule. To make v vote for ω, one could try to create a vicinity for v, in which ω and b are tied. In this case, v would react to the environment, and vote for ω. However, if in the poll ω has any advantage over b (or vice versa), then v would not change her vote, and would continue to vote for a. Of course, if the algorithm knew that ω would beat b by 5 votes, it could easily add 5 more edges for b and create a tie. The problem is that to change the vote of v, the algorithm has to know whether b or ω will lead in the poll and by how much. This is impossible with an unknown, let alone adversarial poll.

Suppose that the size of the poll is at most s. One can try to mitigate the adversarial poll by dividing all the voters who dislike b into $2s$+1 disjoint sets. In set i, the algorithm would assume that the difference in the poll between ω and b is exactly i, and act accordingly. This approach fails, since it only grants ω a fraction $1/(2s+1)$ of the extra votes, which could be very little.

To mitigate the adversarial poll, we create $2s$+1 small gadgets, where in gadget i some voter v would change her mind if the difference in the poll between ω and b is exactly i. We then connect these small gadgets together, to create a cascading effect, which would make most voters vote for ω for any result of the poll. The challenge here is that different voters dislike different candidates, and we need to be concerned with the effects our cascade networks have on each other and on the high degree vertices. The building of theses gadgets is formally described in the proof of the following lemma, which is the unknown poll version of Lemma 1.

LEMMA 4. *Let d and t be natural numbers. Given a desired candidate ω and a voter v such that v does not vote for ω in round 1, it is possible to influence v so that v will support ω from round r, $2s+t+2 \geq r \geq t+2$, even in the unknown poll setting, if:*

1. *v dislikes $a \in C$, $a \neq \omega$, and $d_v \leq d$.*
2. *Every $u \in \mathcal{N}_v$ does not change her vote at least until round $2s+t+2$.*
3. *For each candidate $\ell \in \{a, \omega\}$ there are at least k supporters of ℓ, $k = 2(s+d+1)+(s+t)$.*
4. *There are at least k' voters (other than v) with a degree of at most d who do not vote for ω in round 1 and dislike a, $k' = 2s+t$, and their neighbors do not change their vote at least until round $2s+t+2$.*

PROOF. Let $B^{\bar{a}}$ and $D^{\bar{a}}$ be two sets of voters with a degree of at most d who do not vote for ω in round 1 and dislike a, where $B^{\bar{a}} = \{b_{t+1}, \ldots, b_{2s+t}\}$ and $D^{\bar{a}} = \{d_1, \ldots, d_t\}$. Let $\mathcal{I} = \{v\} \bigcup B^{\bar{a}} \bigcup D^{\bar{a}}$. We call the voters from $B^{\bar{a}}$ *balancers*, since the organizer uses them to cancel out the effect of the poll scores. We call the voters from $D^{\bar{a}}$ *delayers*, since the organizer uses them to delay the round where v changes her vote. The organizer creates a clique between all the voters from $\mathcal{I}$, and connects them to $s+t$ supporters of a. Let $\delta = s^a_{poll} - s^{\omega}_{poll} + s + t$. According to the construction till now, for each $u \in \mathcal{I}$ we can write that $s^a_u - s^{\omega}_u = \delta + (n^a_u - n^{\omega}_u)$ in every round. Clearly, $t \leq \delta \leq 2s+t$, and the organizer can thus assign each possible value of δ (except for the value t) to a voter from $B^{\bar{a}}$, and each value from $\{1, \ldots, t\}$ to a voter from $D^{\bar{a}}$. That is, using Lemma 1 the organizer connects each voter $b_i \in B^{\bar{a}}$ to supporters of a and ω, who will make b_i vote for ω if $\delta = i$. Similarly, the organizer connects each voter $d_i \in D^{\bar{a}}$ to supporters of a and ω, who will make d_i vote for ω if $\delta = i$. Finally, the organizer connects v to supporters of a and ω, who will make v vote for ω if $\delta = 0$.

Now, all the voters vote with their true preferences in round 1. If $t = 0$ and $s^a_{poll} - s^{\omega}_{poll} = -s$ then $\delta = 0$ and v will change her vote to ω in round 2. If $t > 0$ but still $s^a_{poll} - s^{\omega}_{poll} = -s$ then $\delta = t$ and d_t will change her vote in round 2. Since there is an edge between d_t and d_{t-1}, d_{t-1} will change her vote to ω in round 3. Overall, all the voters in $D^{\bar{a}}$ will change their votes to ω one after the other, which will cause v to vote for ω in round $t+2$. If $s^a_{poll} - s^{\omega}_{poll} > -s$ then $\delta \geq 1$ and there is exactly one voter, $b_{\delta+t}$, that will change her vote to ω in round 2. This will cause all the voters from $\{b_{t+1}, \ldots, b_{\delta+t}\}$ and then the voters from $D^{\bar{a}}$ to change their vote to ω one after the other, which will result in v changing her votes to ω in round $\delta+t+2$. Since the voters from $\mathcal{I}$ will change their votes in a row, we call this construction a *chain* and we call the rounds in which they change their vote the *propagation* of the chain. Note that we required that every voter u' that is a neighbor of a voter $u \in \mathcal{I}$ does not change her vote until round $2s+t+2$, since the organizer uses the construction of Lemma 1 and its thus needs to know the exact values of n^a_u and n^{ω}_u. Due to Lemma 1, all the voters from $\mathcal{I}$ will not be affected by their neighbors in G, and all their new neighbors in G' will only increase the score of ω. Therefore, every such voter that will change her vote to ω will not have an incentive to stop supporting ω. □

We will also need to make a specific voter v a supporter, and we thus need to adjust Lemma 2 for the unknown poll setting. The number of supporters will now depend on the poll size:

COROLLARY 5. *Lemma 2 holds for the unknown poll setting with $k = (2s+1)(m-1)(R+1)$.*

PROOF. It is required that v not see any tie between any two candidates i and j, for each possible values of s^i_{poll} and s^j_{poll}. Clearly, we can use Algorithm 1 but replace any $s^{c_i}_v(r)$ with $n^{c_i}_v(r)$. In addition, we will need to change the condition in Line 11 to $|\delta|<s$, which will increase the maximum shift to $(2s+1)(m-1)(R+1)$. □

We are now ready to solve the Influence problem under the unknown poll setting. We use Algorithm 3, which is a variation of Algorithm 2 with the following changes. First, note that the algorithm uses a bijection f, which takes three indexes of candidates as arguments. Therefore, the algorithm refers to all the candidates by their indexes and uses c_ω to denote the desired candidate. Now, instead of using Lemma 2 for influencing type IV voters the algorithm uses Corollary 5. Instead of using Lemma 1 for influencing type II and type III voters the algorithm uses Lemma 4. However, the chain in the proof of Lemma 4 is designed to affect a single voter. In order to affect $O(n)$ voters, the algorithm creates a set of $\sqrt{n}$ chains, denoted SOC, and connects at most $\sqrt{n}$ type II or type III voters to each chain in a SOC. Each chain uses its own sets of balancers and delayers, but all the chains in a SOC use the same sets of supporters and the same number of delayers, which is determined by the bijection f. In addition, the algorithm connects each balancer and delayer to $2\sqrt{n}$ supporters, to cancel any possible impact of the type II or type III voters on the balancers and delayers. Overall, for each $c_\ell, c_i, c_j \in C$, the algorithm creates a SOC that is connected to every type II or type III voter who dislikes c_ℓ and currently votes for c_i, in order to change her vote to c_j. Finally, a new type of voters is defined, type V voters. A voter v is a type V voter if v is in the neighborhood of a balancer or a delayer, v is a bounded degree voter, and v is not already a supporter. Similarly to the type I voters, the algorithm makes every type V voter a supporter by connecting her to a sufficient number of supporters. Overall, we get the same result as with the known poll setting, but with less voters who we are able to affect and more edges that we need to add. Thus:

THEOREM 6. *Let d be a natural number, $z_b=s+d+1$, $z_{B,D}=m^3\sqrt{n}(2s+(m^3-1)/2)$, $R=m^3+z_{B,D}/\sqrt{n}$, $z_{ub}=max((2s+1)(m-1)(R+1), s+d+1)$, $f(\ell,i,j)=m^2(\ell-1)+m(i-1)+j-1$, and $n'=n-(z_{ub}m^2+(d+2)z_{B,D}+2m^3(d+1))$.* INFLUENCE *with unknown poll can be solved in polynomial time if:*

1. *For each $c_\ell, c_i, c_j \in C$ there are k voters with a degree of at most d, who vote for c_ℓ in the first round, and k voters with a degree of at most d, who vote for c_j in the first round, $k=\sqrt{n}(2s+f(\ell,i,j)(s+1))+(d+1)$.*
2. *For each $c_\ell \in C$ there are $z_{ub}m$ voters with a degree of at most d, who vote for c_ℓ in the first round and do not have any voter v in their neighborhood with $d_v > d$.*
3. *For each $c_\ell, c_i, c_j \in C$ there are $\sqrt{n}(2s+f(\ell,i,j)(s+1))$ voters with a degree of at most d who dislike c_ℓ.*
4. *There are no overlaps between the conditions above, i.e., even if a voter meets more than one of the conditions she will only be included in one condition.*
5. *There is at least a fraction p of affected voters with a degree of at most d, where $1-\frac{s}{n'} < p \leq 1$.*
6. *There is at least a fraction p_ω of affected voters with a degree of at most d who do not dislike ω, where $p_\omega > max(\frac{1+(dz_{B,D}+2sm^3\sqrt{n})/n'}{m}, (1-p)+\frac{dz_{B,D}+2sm^3\sqrt{n}}{n'})$.*

PROOF. We use Algorithm 3. To prove correctness, suppose that all the voters in S are supporters. Clearly, all the type I and type V voters will be supporters. According to Corollary 5 all the type IV voters will also be supporters. For using Lemma 4 we need to ensure that the neighbors of the balancers and delayers and of the type II or type III voters will not change their vote until the propagation will end. Indeed, all the neighbors of the balancers and delayers are either type IV voters, type V voters, or voters in S. In addition, the assignment of delayers by f guarantees that all the type II or type III voters from the same SOC will change their vote in the same round, while all the type II or type III voters from different SOCs will change their vote in different rounds. Therefore, all the requirements of Lemma 4 are met and the type II and type III voters will change their vote to c_ω and c_a, respectively. Lemma 4 also guarantees that the type II and type III voters, and all the balancers and delayers, will not change their vote in any subsequent round. We now show that all the voters from S are indeed supporters.

Given a SOC with the chains $C^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$, where $h=f(\ell,i,j)$, each voter $v \in S^\ell[h]$ is connected to at least $\sqrt{n}(2s+h(s+1))+(d+1)$ voters who will vote for c_ℓ. Therefore, the d voters from $\mathcal{N}_v$ and the $\sqrt{n}(2s+h(s+1))$ balancers and delayers of the chains will not have any impact on v's vote. v is also connected to the possibly many affected voters of the chain, who will vote for c_i over the course of several rounds and then will switch to c_j. Since v is not connected to any other voter, v will not have an incentive to change her vote in any round. Using similar argument we conclude that each voter $v \in S^j[h]$ will also not have an incentive to change her vote in any round. Finally, each voter $v \in S^\ell_{(i,i)}$ is connected to z_{ub} voters who vote for c_ℓ, to possibly several type I or type V voter who also vote for c_ℓ, and to at most s type IV voters that do not change their votes. Therefore, v will not have an incentive to change her vote. Thus every $v \in S$ is a supporter.

It remains to show that c_ω will win the election. Clearly, the same argument as in the proof of Theorem 3 can be used, while subtracting the total number of balancers and type V voters from the number of affected voters. □

6. HEURISTIC AND EXPERIMENTAL EVALUATION

In the previous section we provided an algorithm that guarantees a solution will be found for the Influence problem, under some conditions. Using the proposed algorithm, a central election organizer that has access to the preferences of all the voters can easily (in polynomial time) adapt the network to its advantage. However, such an organizer will still be restricted by the number of edges that it is able to add to the social network. The organizer will also want to avoid a situation in which the voters believe there is a "hostile" intervention, and it will thus prefer to minimize the number of voters who will be involved in the adaption of the network. Our algorithm does not take these considerations into account. In this section we present a greedy heuristic for the known poll setting, which is based on our algorithm, that aims to make a desirable candidate ω the winner while minimizing the number of edges that are added to the network, and the number of voters who are involved in the adaption of the network. We use c_{win} to denote the candidate that wins in the first round of the election (when the voters vote according to their true preferences).

6.1 The Heuristic

Similar to Algorithm 2, our heuristic uses some voters as supporters, and causes some of them to change their vote to ω. However, the heuristic does not allocate in advance a full set of supporters $S^\ell_{(i,j)}$ for every $\ell, i, j \in C$, and it does not try to affect all of the other voters. Instead, it allocates only the supporters that are needed, and affects only the minimal number of voters that is required. Thus, the heuristic will prefer to make a voter change her vote to ω if she is voting for c_{win}, since it reduces the gap between the scores of c_{win} and ω by two. It will also prefer to use the same set of supporters

Algorithm 3 Influence (unknown poll)

for all $c_\ell, c_i, c_j \in C$ **do**
  $h \leftarrow f(\ell, i, j)$
  $S^\ell[h] \leftarrow$ a set of k voters with a degree of at most d, who vote for c_ℓ in round 1
  $S^j[h] \leftarrow$ a set of k voters with a degree of at most d, who vote for c_j in round 1
  create a clique between the voters in $S^\ell[h]$
  create a clique between the voters in $S^j[h]$
  for $\mathsf{sc} \leftarrow 1$ to $\sqrt{n}$ **do**
    $B^{\bar{\ell}}_{(i,j)}[\mathsf{sc}] \leftarrow$ a set of $2s$ voters with a degree of at most d, who dislike c_ℓ
    $D^{\bar{\ell}}_{(i,j)}[\mathsf{sc}] \leftarrow$ a set of $h(s+1)$ voters with a degree of at most d, who dislike c_ℓ
    create a chain, $C^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$, according to Lemma 4, using $B^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$, $D^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$, $S^\ell[h]$ and $S^j[h]$
    connect each voter from $B^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$ or $D^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$ to $\sqrt{n}$ voters from $S^\ell[h]$ and $\sqrt{n}$ voters from $S^j[h]$
mark all the chains as "free"
for all $c_\ell, c_i \in C$ **do**
  $S^\ell_{(i,i)} \leftarrow$ a set of z_{ub} voters with a degree of at most d, who vote for c_ℓ in round 1 and do not have any voter v in their neighborhood with $d_v > d$
  create a clique between the voters in $S^\ell_{(i,i)}$
ensure that the sets $S^\ell[h]$, $S^i[h]$, $B^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$, $D^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$, and $S^\ell_{(i,i)}$ are pairwise disjoint
define $S =$ all the voters in the sets $S^\ell[h]$, $S^i[h]$ and $S^\ell_{(i,i)}$
define $B =$ all the voters in the sets $B^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$
define $D =$ all the voters in the sets $D^{\bar{\ell}}_{(i,j)}[\mathsf{sc}]$
for all $c_c \in C$, initialize $s^{c_c}_{curr} \leftarrow s^{c_c}_{truth}$
for all $v \in V \setminus (S \cup B \cup D)$, where $d_v \leq d$ **do**
  define $c_c =$ the candidate that v votes for in round 1
  if $c_c = c_\omega$ **then**
    connect v to z_b voters from $S^\omega_{(\omega,\omega)}$ {*type I voters*}
  else if $v \in \mathcal{N}_u, u \in B \cup D$ **then**
    connect v to z_b voters from $S^c_{(c,c)}$ {*type V voters*}
  else
    $s^{c_c}_{curr} \leftarrow s^{c_c}_{curr} - 1$
    if v does not dislike c_ω **then**
      define $c_a =$ the candidate that v dislikes
      $h \leftarrow f(a, c, \omega)$
      $\mathsf{sc} \leftarrow$ an index of a "free" chain $C^{\bar{a}}_{(c,\omega)}[\mathsf{sc}]$
      make v vote for c_ω, by connecting her to the chain $C^{\bar{a}}_{(c,\omega)}[\mathsf{sc}]$ and to supporters from $S^a[h]$ and $S^\omega[h]$, according to Lemma 4 {*type II voters*}
      if there are $\sqrt{n}$ voters that are connected to $C^{\bar{a}}_{(c,\omega)}[\mathsf{sc}]$ **then**
        mark the chain $C^{\bar{a}}_{(c,\omega)}[\mathsf{sc}]$ as "not-free"
    else
      define $c_a = argmin_{c_\ell \in C \setminus \{c_\omega\}}(s^{c_\ell}_{curr})$
      $h \leftarrow f(\omega, c, a)$
      $\mathsf{sc} \leftarrow$ an index of a "free" chain $C^{\bar{\omega}}_{(c,a)}[\mathsf{sc}]$
      make v vote for c_a, by connecting her to the chain $C^{\bar{\omega}}_{(c,a)}[\mathsf{sc}]$ and to supporters from $S^a[h]$ and $S^\omega[h]$, according to Lemma 4 {*type III voters*}
      if there are $\sqrt{n}$ voters that are connected to $C^{\bar{\omega}}_{(c,a)}[\mathsf{sc}]$ **then**
        mark the chain $C^{\bar{\omega}}_{(c,a)}[\mathsf{sc}]$ as "not-free"
      $s^{c_a}_{curr} \leftarrow s^{c_a}_{curr} + 1$
for all $v \in V$, where $d_v > d$ **do**
  define $c_c =$ the candidate that v votes for in round 1
  make v a supporter, using $S^\ell_{(c,c)}$ for all $\ell \in C$ and Corollary 5 {*type IV voters*}

to affect as many voters as possible. The heuristic is composed of four steps:

1. **Initial stabilization**: The heuristic first checks whether the iterative process converges to ω without adding edges at all (we wait 10 rounds to decide). If so, we are done. Otherwise, the heuristic connects each voter, who would like to change her vote in round 2, to a supporter who makes the voter keep her current vote.
2. **Preparation of potential affected voters**: Let $A^{\bar{i}}_\ell$ be a bag of bounded degree voters, such that $v \in A^{\bar{i}}_\ell$ if v vote for ℓ in the first round and $\omega \succ_v i$. Note that a voter can belong to more than one bag. Now, the heuristic prefers to affect the voters from the bags that can reduce the score of c_{win}, and it also prefers to affect the voters from the bags that are as large as possible. It thus sorts the bags accordingly and then selects the bags, one after the other, till the number of voters in the selected bags is greater than or equals the gap between the scores of c_{win} and ω.
3. **Selection of supporters and affected voters**: Assume $A^{\bar{i}}_\ell$ is the first selected bag. The heuristic selects a voter v from the bag and connects her to supporters of i and ω who will make v believe that i will win unless v will change her vote to ω (similar to Lemma 1). However, the heuristic connects v only to the required number of supporters that is needed, which is usually much lower than our upper bound of $max\Delta$+d_v+1. Then, the heuristic checks whether ω will win the election in round 2 and we can proceed to the next step. If not, the heuristic selects the next voter from the current bag, or if the bag is empty from the next selected bag, and makes her vote for ω, and so on. This step ends when ω wins the election in round 2.
4. **Stabilization**: The heuristic uses Lemma 2 to ensure that all the neighbors of each of the affected voters will not have an incentive to change their vote in round 3.

6.2 Experimental Design

For the empirical evaluation of our heuristic, we used the previously developed social network dataset "Facebook MHRW" [7]. This dataset contains structural information sampled from Facebook, including over 900,000 nodes and the links between them. We sampled our networks using BFS on the dataset, starting from 10 different random nodes as the initial roots, to avoid randomization errors and biases. We ran 100 iterations for each network configuration, and thus every point in our graphs is the average of 1000 runs. We used the following parameters:

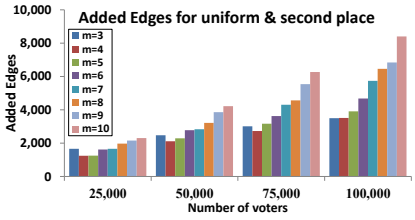
- Number of voters (n): we sampled networks of sizes n=25,000, 50,000, 75,000 and 100,000.
- Number of candidates (m): we used $m = 3, \ldots, 10$.
- The desired candidate (ω): we set ω to be the candidate that reached the 2nd, 3rd, 4th or 5th places according to the true preferences of the voters.
- Preference distribution: we generated the preferences according to 5 distributions that have been examined in the literature [14], with focus on distributions that are claimed to resemble preferences of human societies. Namely, the distributions were a uniform distribution, a uniform single-peaked distribution, and Polya-Eggenberger urn model (with 2-urns and with 3-urns). We also used a dataset of real preference profiles that is available from PrefLib (http://preflib.org). The dataset contains the results of a series of surveys conducted by T.Kamishima asking 5000 individuals their preference about various kinds of sushi.
- Poll size (s): we sampled 500 voters each run to generate the

scores in the poll, which provides a 95% confidence interval for the real scores, given the size of the tested networks.

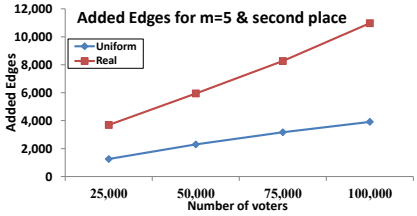
- Maximum degree (d): we used a fixed value of 10, which ensured that less than 5% of the voters had a larger degree.

6.3 Experimental Results

In our first experiment we wanted to check the effect of the number of candidates and the size of the network on the number of added edges. We chose the uniform distribution, which can be easily adapted to different values of m, and we fixed ω to be the candidate in the second place, since presumably the organizer would not try to promote a very unpopular candidate (in the next set of experiments we tested other values of ω). The results are depicted in Figure 2a. As expected, the number of edges increases when we increase the number of voters since the votes of more voters need to change. In addition, when we increase the number of candidates the number of added edges increases, since there are more occurrences of unstable situations, that are solved by adding edges (in step 1 of the heuristic). Still, the number of added edges is not too large compared to the network size. For example, with 5 candidates and a network of 100,000 voters that consist of 366,938 edges we have to add only 3912 edges, which is an increase of 1%. We then fixed the number of candidates to 5, and compared between the uniform and the real distributions. As Figure 2b shows, both distributions had a similar pattern, where the real distribution resulted in adding more edges, but still a reasonable amount: in the network of 100,000 nodes and 366,938 edges we have to add 10,973 edges, which is an increase of 3%.



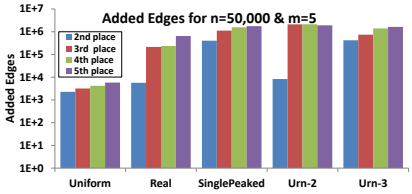
(a) Uniform distribution



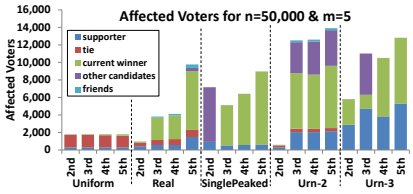
(b) Uniform and real distributions

Figure 2: The number of edges that are added

In the next experiment we wanted to compare the different distributions, and test the behavior of the heuristic when we change ω to be the candidate in the 2nd, 3rd, 4th or even 5th place, while fixing the number of voters (to 50,000) and the number candidates (to 5). The results are depicted in Figure 3a, on a logarithmic scale. As expected, the number of added edges depends on the type of distribution, and when we try to make a a lower placed candidate the winner more edges need to be added. We further explored the number of voters that are given new edges from the heuristic and their types. We divided the affected voters into four types. A "tie" voter is a voter that is connected to one supporter in order to stabilize the network in step 1 of the heuristic. A "current winner" voter is a voter that can reduce the score of c_{win} and is connected to supporters and an "other candidates" voter is a voter that currently does not vote for c_{win}, and is connected to supporters whenever it is needed (in step 3 of the heuristic). Finally, a "friends" voter is a voter that is stabilized in step 4 of the heuristic, using Lemma 2. There are several interesting phenomena that are apparent when comparing Figure 3a and Figure 3b. A comparison of the second place setting in the uniform and real distributions, reveals that in the uniform distribution the number of added edges is lower, while in the real distribution the number of affected voters is higher. The reason for this phenomenon is that the number of voters that need to be stabilized in step 1 of the heuristic is obviously much higher with the uniform distribution, but stabilizing them requires the addition of only one edge. On the other hand, the number of edges that is needed for Lemma 1 with the real distribution is much higher since the gap between the first and second places is bigger. It is also noteworthy that the heuristic surprisingly requires a larger number of affected voters when ω is the second place candidate than when it is the third place candidate, with the single-peaked distribution. This can be explained by observing the types of the voters. When ω is the second place candidate there is an insufficient number of "current winner" voters, and thus a larger number of "other candidates" voters are needed to close the gap between the scores of c_{win} and ω.



(a) Added edges



(b) Added voters

Figure 3: Different distributions and ω's

7. CONCLUSIONS AND FUTURE WORK

We investigated the extent to which social networks affect election outcomes. We used the framework of iterative Plurality voting with an initial poll, and showed that even a small change in the network can lead to a dramatic change in the identity of the winning candidate. We provided a polynomial time algorithm that is able to find the set of edges to add to the network to make a desirable candidate the winner, even if the opinion poll is adversarial and not known in advance. This can be utilized by a central organizer, and we further provided a heuristic that affects the election by adding a minimal number of edges. Using extensive simulations, we demonstrated the effectiveness of the heuristic. In the future, we would like to extend our analysis to other voting rules, and to experimentally examine larger networks with more datasets of real preferences. We would also like to investigate the setting of truth-biased voters, which are voters that revert to their true preferences if they cannot affect the election outcome. Though we have preliminary results for the known poll setting, the unknown poll is much more challenging.

Acknowledgments

This research was supported by the ISRAEL SCIENCE FOUNDATION (grant No. 1488/14), and by ERC Grant #267523.

REFERENCES

[1] N. Alon, M. Babaioff, R. Karidi, R. Lavi, and M. Tennenholtz. Sequential voting with externalities: Herding in social networks. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC-2010)*, pages 36–36, 2012.

[2] L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644, 2011.

[3] S. Chopra, E. Pacuit, and R. Parikh. Knowledge-theoretic properties of strategic voting. In *JELIA*, 2004.

[4] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, 2001.

[5] E. Ephrati, G. Zlotkin, and J. S. Rosenschein. Meet your destiny: A non-manipulable meeting scheduler. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 359–371, 1994.

[6] D. Gatherer. Comparison of eurovision song contest simulation with actual results reveals shifting patterns of collusive voting alliances. *Journal of Artificial Societies and Social Simulation*, 9(2), 2006.

[7] M. Gjoka, M. Kurant, C. T. Butts, and A. Markopoulou. Walking in Facebook: A Case Study of Unbiased Sampling of OSNs. In *Proceedings of IEEE INFOCOM '10*, San Diego, CA, March 2010.

[8] M. Grabisch and A. Rusinowska. Iterating influence between players in a social network. In *Proceedings of the 16th Coalition Theory Network Workshop*, 2011.

[9] U. Grandi, A. Loreggia, F. Rossi, K. Venable, and T. Walsh. Restricted manipulation in iterative voting: Condorcet efficiency and borda score. In *Algorithmic Decision Theory*, volume 8176 of *Lecture Notes in Computer Science*, pages 181–192. 2013.

[10] F. F. Hassanzadeh, E. Y. B. Touri, O. Milenkovic, and J. Bruck. Building consensus via iterative voting. In *IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 1082–1086, 2013.

[11] N. Hazon, R. Lin, and S. Kraus. How to change a group's collective decision? In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI-2013)*, pages 198–205, 2013.

[12] O. Lev and J. Rosenschein. Convergence of iterative voting. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2012)*, pages 611–618, 2012.

[13] A. Maran, N. Maudet, M. Pini, F. Rossi, and K. B. Venable. A framework for aggregating influenced cp-nets and its resistance to bribery. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI-2013)*, pages 668–674, 2013.

[14] R. Meir, O. Lev, and J. Rosenschein. A local-dominance theory of voting equilibria. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation (EC-2014)*, pages 313–330, 2014.

[15] R. Meir, M. Polukarov, J. Rosenschein, and N. Jennings. Convergence to equilibria in plurality voting. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-2010)*, pages 823–828, 2010.

[16] A. Reijngoud and U. Endriss. Voter response to iterated poll information. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2012)*, pages 635–644, 2012.

[17] W. H. Riker and P. C. Ordeshook. A theory of the calculus of voting. *The American Political Science Review*, 62(1):25–42, 1968.

[18] M. Roth, A. Ben-David, D. Deutscher, G. Flysher, I. Horn, A. Leichtberg, N. Leiser, Y. Matias, and R. Merom. Suggesting friends using the implicit social graph. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 233–242, 2010.

[19] L. Sless, N. Hazon, S. Kraus, and M. Wooldridge. Forming coalitions and facilitating relationships for completing tasks in social networks. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2014)*, pages 261–268, 2014.

[20] M. Tal, R. Meir, and Y. Gal. A study of human behavior in online voting. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2015)*, 2015.

[21] H. van Ditmarsch, J. Lang, and A. Saffidine. Strategic voting and the logic of knowledge. In *Proceedings of the 14th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-2013)*, 2013.

[22] T. Walsh. An empirical study of the manipulability of single transferable voting. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI-2010)*, pages 257–262, 2010.

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