False-Name-Proofness in Social Networks

Vincent Conitzer[1], Nicole Immorlica[2], Joshua Letchford[1], Kamesh Munagala[1], and Liad Wagman[3]

[1] Duke University
{conitzer,jcl,kamesh}@cs.duke.edu
[2] Northwestern University
nickle@eecs.northwestern.edu
[3] Illinios Institute of Technology
lwagman@stuart.iit.edu

Abstract. In mechanism design, the goal is to create rules for making a decision based on the preferences of multiple parties (agents), while taking into account that agents may behave strategically. An emerging phenomenon is to run such mechanisms on a social network; for example, Facebook recently allowed its users to vote on its future terms of use. One significant complication for such mechanisms is that it may be possible for a user to participate multiple times by creating multiple identities. Prior work has investigated the design of *false-name-proof* mechanisms, which guarantee that there is no incentive to use additional identifiers. Arguably, this work has produced mostly negative results. In this paper, we show that it is in fact possible to create good mechanisms that are robust to false-name-manipulation, by taking the social network structure into account. The basic idea is to exclude agents that are separated from trusted nodes by small vertex cuts. We provide key results on the correctness, optimality, and computational tractability of this approach.

1 Introduction

Recently, Facebook, Inc. decided to allow its users to vote on its future terms of use [19]. While the result was not binding,[4] this vote represents a new phenomenon that is likely to become more prominent in the future: agents participating in an election or other mechanism through a social networking site. Holding an election among the users of a social networking site introduces some issues that do not appear in regular elections. Perhaps the foremost such issue, and the one that we will focus on, is that it is generally easy for a user to create additional accounts/identities, allowing her to vote multiple times. This can compromise the legitimacy of the election and result in a suboptimal alternative being chosen.

The topic of designing elections or other mechanisms for settings where it is easy to create multiple identities and participate multiple times has already received some attention. The primary approach has been to design mechanisms that are *false-name-proof* [15, 16], meaning that an agent never benefits from participating more than once. (This is analogous to the better-known concept of *strategy-proofness*, meaning that an agent never benefits from misreporting her preferences. In fact, false-name-proofness is often defined in a way that subsumes strategy-proofness.) Unfortunately, existing results on false-name-proofness are quite negative, especially in voting contexts. For the

[4] The result would have been binding if at least 30% of all active users had voted, a seemingly impossibly high turnout in this context.

case where additional identities can be created at zero cost, a general characterization of false-name-proof voting mechanisms has been given [5]; this characterization implies that for the special case where there are only two alternatives, the best we can do is the *unanimity* mechanism. This mechanism works as follows: if all voters agree on which alternative is better, that alternative is chosen; but if there is any disagreement (no matter in which proportions), then a fair coin is flipped to decide between the alternatives. This is an extremely negative result, since the mechanism is almost completely unresponsive to the votes.[5] Several ways to circumvent such negative results have been proposed, such as assuming that creating additional identities comes at a small cost [14] or considering a model in which it is possible to verify some of the identities [4].

These prior results do not consider any social network structure that may hold among the identities. Rather, these earlier results can be thought of as applying to settings where a user creates an account for the sole purpose of casting a vote (or bid, etc.), so that no social network structure is specified. We will show in this paper that by using the social network structure in the mechanism, it is possible to obtain much more positive results, because fake identities will look suspect in the social network (graph) structure. To give some intuition, consider John Doe, who has a legitimate account on the social networking site. In order to cast more votes, he can create several other identities (*false names*), such as Jane Jones and Jimmy Smith. Among the accounts that he controls, he can create any network structure by linking them to each other. However, if the other users behave legitimately, then he will not be able to link his additional accounts to any of the other users' identities (since, after all, they have never heard of Jane Jones or Jimmy Smith); he will only be able to get his friends to link to his legitimate identity (John Doe). This results in an odd-looking social network structure, where his legitimate identity constitutes a vertex cut in the graph, whose removal separates the fake identities from the rest of the graph.

In the remainder of this paper, we generalize the intuition afforded in the above scenario, giving a notion of when a node is "suspect" based on small vertex cuts that separate it from the trusted nodes. In Section 2, we formally define the setting that we will focus on. In Section 3, we discuss false-name-proofness and provide a sufficient condition for guaranteeing it. In Section 4, we discuss how to find all suspect nodes when trusted nodes are given exogenously to the algorithm. Then, in Section 5, we extend our analysis to settings in which we do not have trusted nodes initially, but we can actively verify nodes. We give both correctness and optimality results. The full version of this paper includes all the proofs and some additional examples, as well as simulation results for random graph models, in which we investigate how many vertices will typically be regarded as suspect (exogenous case) or how many need to be verified (endogenous case).

[5] The literature on false-name-proof voting mechanisms is quite recent: earlier work on false-name proofness considered other settings, such as *combinatorial auction* mechanisms, where multiple items are for sale at the same time. Unfortunately, here, too, there are strong impossibility results, including a result that states that under certain conditions, from the perspective of a worst-case efficiency ratio, it is impossible to significantly outperform the simple mechanism that sells all items as a single bundle [8].

Related Work. The basic intuition that the creation of false identities in a social network results in suspiciously small vertex cuts has previously been explored in several papers, in peer-to-peer networks [18, 17] and web spam detection [2, 3, 6, 7, 13].

The work on fraud in peer-to-peer networks attempts to thwart Sybil attacks in which one or more malicious users obtain multiple identities in order to out-vote legitimate users in collaborative tasks like Byzantine failure defenses. These papers propose protocols that ensure that *not too many* false identities are accepted. While this may be sufficient to thwart certain Sybil attacks in decentralized distributed systems, it can still leave incentives for an agent to create multiple identities, especially in applications such as elections in which the electorate is about evenly divided. Furthermore, a major hurdle in the Sybil attack research is that any protocol must be decentralized. In contrast, in this paper, we follow the stricter approach of guaranteeing that the creation of false identities is always weakly suboptimal, corresponding to the standard approach in the mechanism design literature. On the other hand, we allow our mechanisms to be centralized, as we envision them being run by the proprietor of the social network who has access to the network structure.

Fraud is also prevalent in the world wide web where users sometimes create fake webpages and links with the sole intent of boosting the PageRank of given website(s). Several researchers have considered using link structure to combat spam [2, 3, 6, 7, 13]. In SpamRank [2, 3], the authors assume that a node is suspect if the main contribution to its PageRank is generated from a small set of supporting nodes (see also [6]). Our focus on small vertex cuts can be interpreted as an extreme version of the conditions proposed in SpamRank. An alternative approach, as taken by TrustRank [7] and Anti-TrustRank [13], assumes the existence of an oracle (e.g., a human being) which is able to determine the legitimacy of any given website. Calls to the oracle are, however, expensive, and so the main task in the protocol is to select a seed set of pages. The protocol then guesses the legitimacy of the remaining pages based on their connectivity to the seed set. In particular, the protocol assumes that legitimate pages rarely point to illegitimate ones, and hence the illegitimate pages are those that are "approximately isolated." Again, this approach is similar to our approach at a high level; the selection of the seed set corresponds to our verification policy (discussed later in the paper), and the condition of approximate isolation corresponds to the condition of small vertex cuts in our work. Despite these similarities, the particulars of the model and definitions are quite different, as these protocols are designed to combat fraudulent attacks in PageRank, whereas our goal is to prevent fraudulent attacks in voting or other mechanisms.

2 Setting

Our results can be applied to any mechanism design domain, but for the sake of concreteness, it may be helpful to think about the simple setting in which m agents must select between two alternatives. Each agent has a strict preference for one alternative over the other. The mechanism designer wishes to make a socially desirable choice, i.e., select an alternative that is beneficial for society as a whole. The majority rule, in which the alternative preferred by more voters wins, would be ideal; unfortunately, the majority rule will result in incentives to create false names, if naïvely applied.

Agents are arranged in a social network consisting of n nodes where $m \leq n$. Each agent i has a legitimate account in the social network, corresponding to a node v_i^t, as well as a (possibly empty) set of illegitimate accounts V_i^f. There is an arbitrary graph structure among the legitimate nodes in the social network—that is, we impose no structure on the subgraph induced by the legitimate nodes $\{v_i^t\}_{i \in \{1,\ldots,m\}}$.

In the most basic version of our model, we assume that no two manipulating agents can work together, so that an agent can only link her illegitmate nodes to each other and to her own legitimate node. Hence, for any $i \neq j$, there are no edges between V_i^f and $\{v_j^t\} \cup V_j^f$. However, for each agent i, we allow an arbitrary graph structure on $\{v_i^t\} \cup V_i^f$.

In the more general version of our model, we assume that up to k agents can collude together. (The basic model is the special case where $k = 1$.) That is, the agents $1, \ldots, m$ are partitioned into coalitions $S_j \subseteq \{1, \ldots, m\}$, with $|S_j| \leq k$ for each j. Let $V_{S_j}^f$ be the set of all illegitimate nodes used by S_j, that is, $V_{S_j}^f = \bigcup_{i \in S_j} V_i^f$, and let $V_{S_j}^t$ be the set of all legitimate nodes used by S_j, that is, $V_{S_j}^t = \bigcup_{i \in S_j} \{v_i^t\}$. Two distinct coalitions cannot link their illegitimate nodes to each other, so that for any $i \neq j$, there are no edges between $V_{S_i}^f$ and $V_{S_j}^t \cup V_{S_j}^f$. However, for each coalition S_i, we allow an arbitrary graph structure on $V_{S_i}^t \cup V_{S_i}^f$.

To summarize, our social network setting consists of

- a set of m agents denoted $\{1, \ldots, m\}$,
- a set of m legitimate nodes, one for each agent, denoted $V^t = \{v_1^t, \ldots, v_m^t\}$,
- a collection of m (possibly empty) sets of illegitimate nodes, one for each agent, denoted $\{V_1^f, \ldots, V_m^f\}$,
- a partition of the agents $\{1, \ldots, m\}$ into subsets S_j, where $|S_j| \leq k$ (the no-collusion case corresponds to $k = 1$), such that for any i, j, there are no edges between $V_{S_i}^f$ and $V_{S_j}^t \cup V_{S_j}^f$ (apart from this, the graph structure can be arbitrary).

Some of the nodes in the graph will be *trusted*. For example, the mechanism designer may personally know the agents corresponding to these nodes in the real world. This is a case in which trust is *exogenous*, that is, we have no control over which agents are trusted: the trusted agents are given as part of the input. Later in the paper, we will consider settings where we can, with some effort, *verify* whether any particular node is legitimate (for example, by asking the node for information that confirms that there is a corresponding agent in the real world). Nodes that pass this verification step become trusted nodes; this is a case of *endogenous* trust. It should be noted that, in either case, we do *not* assume that a trusted node will refrain from creating additional identifiers. That is, the only sense in which the node is trusted is that we know it corresponds to a real agent.

The mechanisms that we consider in this paper operate as follows. A *suspicion policy* is a function that takes as input the social network graph $G = (V, E)$ as well as a set T of trusted nodes, $T \subseteq V^t \subseteq V$; and as output labels every node in V as either "deemed legitimate" or "suspect." Generally, all the nodes in T will be deemed legitimate, but others may be deemed legitimate as well based on the network structure. Subsequently, all the nodes that have been deemed legitimate get to participate (*e.g.*,

vote) in a standard mechanism f (*e.g.*, the majority rule), and based on this an outcome is chosen. (In this context, we only consider *anonymous* mechanisms that treat all nodes that get to participate identically.) In the case where nodes become trusted through verification, we also have a *verification policy* that takes G as input and determines which nodes to verify.

We consider a game played between the mechanism designer and the agents (more precisely, the coalitions S_j). First, the mechanism designer announces her mechanism, consisting of f and the suspicion policy (and, in the case where trust is obtained through verification, a verification policy). Then, each coalition S_j creates its illegitimate nodes $V^f_{S_j}$, as well as the edges that include these nodes (they can only have edges to other nodes in $V^f_{S_j}$, and to $V^t_{S_j}$). Note that the coalitions do *not* strategically determine edges between legitimate nodes in this game: in order to focus on false-name manipulation, only the creation of false nodes and their edges is modeled in the game. Also note that the mechanism designer, when announcing her mechanism, is unaware of the true graph as well as which agents are in coalitions together.

After obtaining the social network graph (and, possibly, some exogenously trusted nodes), the mechanism designer runs (1) (possibly) the verification policy and (2) the suspicion policy. The designer subsequently asks the nodes that have been deemed legitimate to report their preferences, and then finally runs (3) the standard mechanism f on these reported preferences, to obtain the outcome.

Whether this results in incentives for using false names depends on all of the components (1), (2), and (3), and each one individually can be used to make the whole mechanism false-name-proof. For example (for component 3), if f is by itself false-name-proof, then even if we verify no nodes and deem every node legitimate, there is still no incentive to engage in false-name manipulation. The downside of this approach is that we run into all the impossibility results from the literature on designing false-name-proof mechanisms. Similarly (for component 1), if we verify all nodes and then only deem the trusted nodes (the ones that passed the verification step) legitimate, there is no incentive to use false names. Of course, this generally results in far too much overhead. In this paper, we will be interested in suspicion policies (component 2) that by themselves guarantee that there is no incentive to use false names. For this, we heavily rely on the social network structure. In the first part of the paper, we do not consider verification policies—we take which nodes are trusted as given exogenously.

3 False-name-proofness

To define what it means for a suspicion policy to guarantee false-name-proofness, we first need to define some other properties. The next two definitions assume that a coalition can be thought of as a single player with coherent preferences; this is reasonable in the sense that if there is internal disagreement within the coalition, this will only make it more difficult for them to manipulate the mechanism.

Definition 1. *A standard mechanism f is* k-strategy-proof *if it is a dominant strategy for every coalition of size at most k to report truthfully.*

Definition 2. *A standard (anonymous) mechanism f* satisfies k-voluntary participation *if it never helps a coalition of size at most k to use fewer identifiers.*

Because the coalitions play a game with multiple stages, it is important to specify what we assume the coalitions learn about each other's actions in earlier stages—that is, what are the information sets in the extensive form of the game? Specifically, when a coalition reports its preferences to f, what does the coalition know about the nodes and edges created by other coalitions? We assume that a coalition learns nothing about other coalitions' actions, except that the coalition can (possibly) make inferences about what others have done based on which of its own nodes have been deemed legitimate. Thus, it is assumed that each coalition is rational and has perfect recall, but also that it does not have any other way of observing what other coalitions have done.

Definition 3. *We say that the* Limited Information Assumption (LIA) *holds if, for every coalition S_j, for every two nodes*[6] *ν_1, ν_2 in the extensive form of the game (where S_j is about to report preferences to f), the following holds. If S_j has taken the same node-and-edge creation actions at ν_1 and ν_2, and the same nodes have been deemed legitimate for S_j at ν_1 and ν_2, then these nodes are in the same information set—that is, S_j cannot distinguish them.*

It should be emphasized that LIA does not specify the information sets exactly—it is merely an *upper bound* on how much the coalitions learn about each other's actions. Specifically, we can also require the coalitions to report preferences for nodes *before* informing them exactly which of these nodes have been deemed legitimate. In an extreme special case of this (for which our results still hold), we can consider the situation where a coalition must create nodes and edges and report preferences for its nodes *at the same time*, making the game a single-stage game. In this case, when a coalition is reporting preferences, it clearly knows nothing about what the other coalitions have done at all, since they are moving at the same time. This is equivalent to saying that a coalition first creates nodes and edges, and then reports preferences for these nodes but without learning anything (including which of these nodes have been deemed legitimate). This is consistent with LIA: it just means that even more nodes in the game tree are in the same information set than is strictly required by LIA.

We now define what it means for a suspicion policy to guarantee false-name-proofness.

Definition 4. *A suspicion policy Π* guarantees false-name-proofness for coalitions of size at most k *if, under the LIA assumption, the following holds. For any standard (anonymous) mechanism f that is k-strategy-proof and satisfies k-voluntary participation, if we combine Π with f, then for any true social network structure on V^t, for any initial trusted nodes $T \subseteq V^t$, and for any partition of V^t into coalitions S_j of size at most k each, it is a dominant strategy for each coalition to set $V^f_{S_j} = \emptyset$ and report truthfully.*

A Sufficient Condition for Guaranteeing False-Name-Proofness. We now provide a sufficient condition for guaranteeing false-name-proofness.

Definition 5. *A suspicion policy Π is* k-robust *if, for any true social network structure on V^t, for any initial trusted nodes $T \subseteq V^t$, and for any partition of V^t into coalitions S_j of size at most k each, we have the following. For every coalition S_j, for every profile of actions taken by the other coalitions:*

[6] These are not to be confused with the nodes in the network.

1. *The actions of S_j (in terms of creating new nodes and edges) do not affect which of the other coalitions' identifiers ($V \setminus (V^t_{S_j} \cup V^f_{S_j})$) are deemed legitimate.*
2. *The number of identifiers in $V^t_{S_j} \cup V^f_{S_j}$ that are deemed legitimate is maximized by setting $V^f_{S_j} = \emptyset$.*

Theorem 1. *If a suspicion policy Π is k-robust, then it guarantees false-name-proofness for coalitions of size at most k.*

4 Exogenously Given Trusted Nodes

We begin by studying the case where the trusted nodes T are given exogenously. This could correspond to the case where the mechanism designer personally knows the owners of some of the nodes on the network, or perhaps these nodes have already been successfully verified in an earlier stage. Later in the paper, we will study the case where there are no exogenously given trusted nodes, so that we have to decide which nodes to verify. Given G and T, the next step is to determine which nodes to label as "suspect," based on the fact that they are not well connected to trusted nodes. We will make our suspicion policy precise shortly, but first we illustrate the basic idea on a small example. We recall that k denotes the maximum size of a coalition of colluding agents. Figure 1 gives an example of a network with two exogenously given trusted nodes, for the case where $k = 1$. As the figure illustrates, nodes that are separated from the trusted nodes

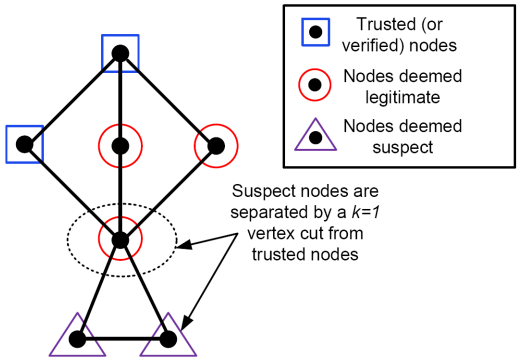

Fig. 1. Example network. The nodes correspond to identities (user accounts), and the edges correspond to (say) friendship relations between the identities. The mechanism designer, at this point for exogenous reasons, considers certain nodes "trusted" (marked by squares), that is, she is sure that they are not false names. The nodes marked with triangles are separated from the trusted nodes by a vertex cut of size one (indicated by the dotted ellipse). As a result, it is conceivable that these nodes are false names, created by the agent corresponding to the vertex-cut node; hence, they are labeled *suspect*. The remaining nodes are not separated from the trusted nodes by a vertex cut of size one, and as a result they are deemed *legitimate* (marked by circles).

by a vertex cut of size 1 could be false identities created by the node on the vertex cut in order to manipulate the outcome of the mechanism. Hence, they are deemed *suspect*.

In the following subsections, we first define our suspicion policy precisely and prove that it has several nice properties, including guaranteeing false-name-proofness. We then prove that this policy is optimal in the sense that any other suspicion policy with these properties would label more nodes as suspect. Finally, we give a polynomial-time algorithm for determining whether nodes are deemed legitimate or not under this policy.

The Suspicion Policy. One natural approach is to label as suspect every node v that is separated from all the trusted nodes by a vertex cut of size at most k (this cut may include some of the trusted nodes). After all, such a node v may have been artificially created by a coalition of nodes corresponding to its vertex cut. On the other hand, for a node v that is *not* separated from the trusted nodes by any vertex cut of size at most k, there is no coalition of nodes that could have artificially created v. While this reasoning is correct, it turns out that, to guarantee false-name-proofness, it is not sufficient to label *only* the nodes separated from the trusted nodes by a vertex cut of size at most k as suspect. The reason is that this approach may still leave an incentive for a coalition to create false nodes: not because these false nodes will be deemed legitimate, but rather because it may prevent *other* nodes from being labeled as suspect. We first observe a fundamental property of nodes being separated from the trusted nodes by a vertex cut of size at most k.

Lemma 1 (cf. Menger [11]). *For an initially untrusted node v, the following two statements are equivalent.*

1. *v is not separated from the initially trusted nodes by a vertex cut of size at most k (which may include initially trusted nodes).*
2. *There exist $k+1$ vertex-disjoint paths from (distinct) initially trusted nodes to v.*

The problem with the approach above is that a coalition may use false nodes that will be labeled suspect, but that help create paths to other nodes that will be deemed legitimate as a result. The solution is to apply the procedure *iteratively*, in each stage removing the nodes that are separated from all the trusted nodes by a vertex cut of size at most k, until convergence.

Definition 6. *Let r take as input $G = (V, E)$ and $T \subseteq V$, and as output produce the subgraph G' of G that results from removing those nodes in $V - T$ that are separated from the trusted nodes T by a vertex cut of size at most k (as well as removing the edges associated with these nodes). These vertex cuts are allowed to include nodes in T. Let $G = G^{(0)}, G' = G^{(1)}, G^{(2)}, \ldots, G^{(n_{G,T})}$ be the sequence of graphs that results from applying r iteratively on $(G^{(i)}, T)$, where $n_{G,T}$ is the smallest number satisfying $G^{(n_{G,T})} = G^{(n_{G,T}-1)}$ (note this sequence must converge as the set of nodes in successive iterations is nonincreasing). Then our suspicion policy Π_k^*, when applied to (G, T), deems all the nodes in $G^{(n_{G,T})}$ legitimate, and all the other nodes in G suspect.*

In each iteration, the procedure for computing Π_k^* removes *all* the nodes that are at that point separated from all the trusted nodes by a vertex cut of size at most k. This

corresponds to eliminating nodes in a particular order. One may wonder if the result would be any different if we eliminated nodes in a different order, for example, in one iteration removing only a subset of the nodes that are at that point separated from all the trusted nodes by a vertex cut of size at most k, before continuing to the next iteration. This is analogous to the notion of path independence of iterated strict dominance in game theory: no matter in which order we eliminate strictly dominated strategies, in the end we obtain the same set of remaining strategies [10]. (This is in contrast to iterated *weak* dominance, where the order of elimination does affect the final remaining strategies.) We will show a similar path independence result for removing nodes in our setting. To do so, we first define the class of suspicion policies that correspond to *some* order; then we show that the class has only one element, namely, Π_k^*.[7]

Definition 7. *Let $\mathbf{\Pi}_k$ be the class of all suspicion policies that correspond to a procedure where:*

- *In each iteration, some subset of the nodes that are at that point separated from all the trusted nodes by a vertex cut of size at most k is eliminated from the graph;*
- *This subset must be nonempty when possible;*
- *When no additional nodes can be eliminated, the remaining nodes are exactly the ones deemed legitimate.*

Lemma 2. *The class $\mathbf{\Pi}_k$ consists of a singleton element Π_k^*,* i.e., *$\mathbf{\Pi}_k = \{\Pi_k^*\}$.*

We now show that our policy Π_k^* guarantees false-name-proofness for coalitions of size at most k.

Lemma 3. *Let $G = (V, E)$ be a graph and let $T \subseteq V$ be the trusted nodes. Let G' be a graph that is obtained from G by adding additional nodes V' and additional edges E' that each have at least one endpoint in V'—in such a way that every node in V' is separated from T by a vertex cut of size at most k. Then, applying Π_k^* to $G' = (V \cup V', E \cup E')$ and T results in the same nodes being deemed legitimate as applying Π_k^* to G and T.*

Theorem 2. *Π_k^* is k-robust (and hence, by Theorem 1, guarantees false-name-proofness for coalitions of size at most k). Moreover, under Π_k^*, a coalition S_j's actions also do not affect which of its own legitimate nodes $V_{S_j}^t$ are deemed legitimate. Finally, Π_k^* is guaranteed to label every illegitimate node as suspect.*

Optimality. We now show that Π_k^* is the best possible suspicion policy in the sense that any other policy satisfying the desirable properties in Theorem 2 must label more nodes as suspect.

Theorem 3. *Let Π' be a suspicion policy that (1) is k-robust, (2) is such that a coalition S_j's actions also do not affect which of its own legitimate nodes $V_{S_j}^t$ are deemed legitimate, and (3) is guaranteed to label every illegitimate node as suspect. Then, if Π_k^* labels a node as suspect, then so must Π'.*

[7] The different orders of course correspond to different *procedures* for computing which nodes are deemed legitimate, but we will show that as a *function* that determines which nodes are finally deemed legitimate, they are all the same.

Polynomial-time Algorithm for Determining Whether a Node is Suspect. In this subsection, we give a polynomial-time algorithm for determining whether nodes are deemed legitimate or suspect according to Π_k^*. The key step is to find an algorithm for figuring out which nodes are separated from the trusted nodes by a vertex cut of size at most k; then we can simply iterate this in order to execute Π_k^* (and by Lemma 2 we do not need to be careful about the order in which we eliminate nodes). It turns out that by Lemma 1, we can do this by solving a sequence of maximum flow problem instances.

Theorem 4. *Given $G = (V, E)$ and $T \subseteq V$, we can determine in polynomial time which nodes are not separated from T by a vertex cut of size at most k. As the number of iterations of Π_k^* is bounded by $|V|$, we can run Π_k^* in polynomial time.*

5 Choosing Nodes to Verify (Endogenous Trust)

Our methodology requires some nodes to be trusted. So far, we have considered settings where some nodes are trusted for exogenous reasons (for example, the organizer's own friends may be the only trusted nodes). However, we can also endogenize which nodes are trusted, by assuming that the organizer can invest some effort in *verifying* some of the identities to establish their legitimacy (for example, by asking these identities for information that identifies them in the real world). This is an approach that has been considered before in the context of false-name-proofness [4], but that prior work paid no regard to social network structure. The social network structure can drastically reduce the amount of verification required, because, as we have seen earlier in this paper, once we have some nodes that are trusted, we can infer that others are legitimate.

There are (at least) two approaches to consider here: verify enough nodes so that no suspect nodes remain at all (and try to minimize the number of verified nodes under this constraint), or try to maximize the number of nodes deemed legitimate, given a budget of verifications (say, at most b verifications). In this paper, we focus on the former.

Technically, a verification policy consists of a contingency plan, where the next node to verify depends on the results of earlier verifications of nodes (which can either fail or succeed). If a node fails the verification, that node is classified as illegitimate, and the verification continues. The verification continues until no nodes remain suspect (other than ones that failed the verification step)—that is, until no unverified nodes are separated by a vertex cut of size at most k from the nodes that were successfully verified. (This vertex cut can include successfully verified nodes. We note that in this context there is no longer a reason to *iteratively* remove nodes in the procedure that computes the trust policy (Π_k^*): because our goal is for *all* remaining nodes to be deemed legitimate, we simply need to check whether any nodes are removed in the first iteration.)

Optimally Deciding Which Nodes to Verify. We now turn to the following optimization problem: how do we minimize the number of nodes that we verify before reaching the point where all the remaining nodes are deemed legitimate? To answer this question, we first note that, since there will be no incentive to create illegitimate nodes, we can assume that all nodes will in fact be legitimate. (This does not mean that we can afford to not do the verification, because if we did not, then there would be incentives to create illegitimate nodes again.) Hence, the problem becomes to find a minimum-size subset of nodes so that no other node is separated from these nodes by a vertex cut of size at most k (which may include nodes in this subset)—or, equivalently, by Lemma 1,

to find a minimum-size subset of nodes so that every other node is connected by $k+1$ vertex-disjoint paths to (distinct nodes in) this subset.

This problem is a special case of the *source location* problem. A polynomial-time algorithm for this problem is given in a paper by Nagamochi et al. [12]. They show that the problem has a matroidal property, as follows. Instead of thinking about minimizing the number of verified nodes, we can think about maximizing the number of unverified nodes. Say a subset $U \subseteq V$ is *feasible* if, for every $v \in U$, there exist $k+1$ vertex-disjoint (apart from v) paths to (distinct) nodes in $V \setminus U$.

Theorem 5 ([12]). *The feasible sets satisfy the independence axioms of a matroid.*

Finding an independent set of maximum size in a matroid is easy: start with an empty set, and attempt to include the elements one at a time, being careful not to violate the independence property. In the context of trying to find a minimum-size set of nodes to verify, this corresponds to starting with the set of *all* nodes, and attempting to *ex*clude the nodes one at a time, being careful that it will still result in all the excluded nodes being deemed legitimate. To check the latter, we only need to consider the current node:

Lemma 4. *Suppose $S \subseteq V$ is such that from every $u \in V - S$, there exist $k+1$ vertex-disjoint paths to (distinct nodes in) S, and suppose that for some v, $S - \{v\}$ does not have this property. Then, there do not exist $k+1$ vertex-disjoint paths from v to (distinct nodes in) $S - \{v\}$.*

This results in the following simple polynomial-time algorithm Φ_k for finding a minimum-size set of nodes to verify.

Definition 8. *Φ_k takes as input a graph $G = (V, E)$ and proceeds as follows to determine the nodes S to verify:*

1. *Initialize $S \leftarrow V$.*
2. *For each node $v \in S$: if there are $k+1$ vertex-disjoint paths from $S - \{v\}$ to v, then remove v from S.*
3. *Return S.*

6 Conclusions and Future Research

From the above, it becomes clear that false-name-proofness, while achievable in social networking settings, does not come for free: we either cannot let all agents participate, or we must spend significant effort verifying identities. How severe these downsides are depends on the exact structure of the social network. If we have a sufficiently densely connected social network, then almost everyone can participate even when there are relatively few trusted identities, or, alternatively, we only need to verify a small number of identities to let everyone participate. But, is this likely to be the case in realistic social networks? The full version of our paper has some simulation results. Future research may also be devoted to considering some changes in the basic model and their effect on our results. What happens if agents can decide to drop edges (that is, not declare friendships) for strategic reasons? What happens if agents can get other agents to link to their fake identities at a cost? Results here may be reminiscent of those obtained in existing models where additional identifiers can be obtained at a cost [14]. What

happens when we can only verify a limited number of nodes and try to maximize the number of nodes deemed legitimate?

Acknowledgements. Conitzer and Letchford were supported by NSF under award numbers IIS-0812113 and CAREER 0953756, and by an Alfred P. Sloan Research Fellowship. Munagala was supported by an Alfred P. Sloan Research Fellowship, and by NSF via CAREER award CCF-0745761 and grants CCF-1008065 and CNS-0540347. Wagman was supported by an IIT Stuart School of Business Research Grant.

References

1. Réka Albert and Albert–László Barabási. Statistical mechanics of complex networks, 2001.
2. R. Andersen, C. Borgs, J. Chayes, J. Hopcroft, V. Mirrokni, and S. Teng. Local computation of PageRank contributions. In *WAW*, 2007.
3. Reid Andersen, Christian Borgs, Jennifer T. Chayes, John E. Hopcroft, Kamal Jain, Vahab S. Mirrokni, and Shang-Hua Teng. Robust PageRank and locally computable spam detection features. In *AIRWeb*, pages 69–76, 2008.
4. Vincent Conitzer. Limited verification of identities to induce false-name-proofness. In *TARK*, pages 102–111, Brussels, Belgium, 2007.
5. Vincent Conitzer. Anonymity-proof voting rules. In *Proceedings of WINE*, pages 295–306, Shanghai, China, 2008.
6. Zoltán Gyöngyi, Pavel Berkhin, Hector Garcia-Molina, and Jan Pedersen. Link spam detection based on mass estimation. In *Proceedings of VLDB*. ACM, 2006.
7. Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with TrustRank. In *Proceedings of VLDB*, pages 576–587. Morgan Kaufmann, 2004.
8. Atsushi Iwasaki, Vincent Conitzer, Yoshifusa Omori, Yuko Sakurai, Taiki Todo, Mingyu Guo, and Makoto Yokoo. Worst-case efficiency ratio in false-name-proof combinatorial auction mechanisms. In *Proceedings of AAMAS*, pages 633–640 Toronto, Canada, 2010.
9. Jon M. Kleinberg. Navigation in a small world. *Nature*, 2000.
10. Duncan R. Luce and Howard Raiffa. Games and decisions: introduction and critical survey, New York, 1957
11. Karl Menger. Zur allgemeinen Kurventheorie. *Fund. Math.*, 10:96–115, 1927.
12. Hiroshi Nagamochi, Toshimasa Ishii, and Hiro Ito. Minimum cost source location problem with vertex-connectivity requirements in digraphs. *Information Processing Letters*, 80(6):287–293, 2001.
13. R. Raj and V. Krishnan. Web spam detection with anti-trust rank. In *AIRWeb*, pages 381–389, 2006.
14. Liad Wagman and Vincent Conitzer. Optimal false-name-proof voting rules with costly voting. In *Proceedings of AAAI*, pages 190–195, Chicago, IL, USA, 2008.
15. Makoto Yokoo, Yuko Sakurai, and Shigeo Matsubara. Robust combinatorial auction protocol against false-name bids. *Artificial Intelligence*, 130(2):167–181, 2001.
16. Makoto Yokoo, Yuko Sakurai, and Shigeo Matsubara. The effect of false-name bids in combinatorial auctions: New fraud in Internet auctions. *GEB*, 46(1):174–188, 2004.
17. Haifeng Yu, Phillip B. Gibbons, Michael Kaminsky, and Feng Xiao. SybilLimit: A near-optimal social network defense against sybil attacks. *ToN*, 18(3):885-898, 2010.
18. Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. SybilGuard: Defending against sybil attacks via social networks. *ToN*, 16(3):576–589, 2008.
19. Mark Zuckerberg. Voting begins on governing the Facebook site, 2009. http://blog.facebook.com/blog.php?post=76815337130.