Graph-Based Fraud Detection in the Face of Camouflage

BRYAN HOOI, KIJUNG SHIN, HYUN AH SONG, ALEX BEUTEL, NEIL SHAH, and CHRISTOS FALOUTSOS, Carnegie Mellon University

Given a bipartite graph of users and the products that they review, or followers and followees, how can we detect fake reviews or follows? Existing fraud detection methods (spectral, etc.) try to identify dense subgraphs of nodes that are sparsely connected to the remaining graph. Fraudsters can evade these methods using *camouflage*, by adding reviews or follows with honest targets so that they look "normal." Even worse, some fraudsters use *hijacked accounts* from honest users, and then the camouflage is indeed organic.

Our focus is to spot fraudsters in the presence of camouflage or hijacked accounts. We propose FRAUDAR, an algorithm that (a) is camouflage resistant, (b) provides upper bounds on the effectiveness of fraudsters, and (c) is effective in real-world data. Experimental results under various attacks show that FRAUDAR outperforms the top competitor in accuracy of detecting both camouflaged and non-camouflaged fraud. Additionally, in real-world experiments with a Twitter follower–followee graph of 1.47 billion edges, FRAUDAR successfully detected a subgraph of more than 4,000 detected accounts, of which a majority had tweets showing that they used follower-buying services.

CCS Concepts: • **Information systems** → **Social networks**; *Content ranking;*

Additional Key Words and Phrases: Fraud detection, link analysis, spam detection

ACM Reference Format:
Bryan Hooi, Kijung Shin, Hyun Ah Song, Alex Beutel, Neil Shah, and Christos Faloutsos. 2017. Graph-based fraud detection in the face of camouflage. ACM Trans. Knowl. Discov. Data 11, 4, Article 44 (June 2017), 26 pages.
DOI: http://dx.doi.org/10.1145/3056563

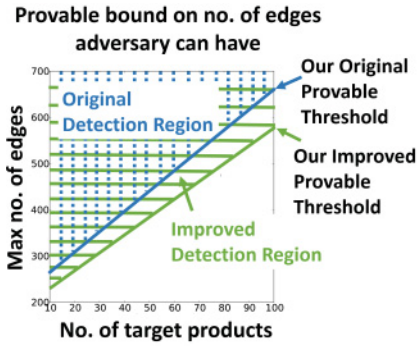
1. INTRODUCTION

How can we detect if a politician has purchased fake followers on Twitter, or if a product's reviews on Amazon are genuine? More challengingly, how can we *provably* prevent fraudsters who sell fake followers and reviews for various web services from evading our detection systems? In this article, we focus on precisely this problem—specifically, how can we design a fraud detection system with strong, provable guarantees of robustness?

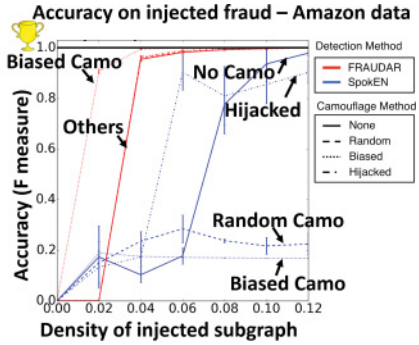
Given the rise in the popularity of social networks and other web services in recent years, fraudsters have strong incentives to manipulate these services. On several shady websites, anyone can buy fake Facebook page-likes or Twitter followers by the thousands. Yelp, Amazon and TripAdvisor fake reviews are also available for sale,

This material is based upon work supported by the National Science Foundation under Grant No. CNS-1314632, DGE-1252522, and IIS-1408924.
Authors' addresses: B. Hooi, Machine Learning Department and Department of Statistics, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 USA; email: bhooi@andrew.cmu.edu; H. A. Song, Machine Learning Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 USA; email: hyunahs@cs.cmu.edu; A. Beutel, N. Shah, K. Shin, and C. Faloutsos, Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 USA; emails: {abeutel, neilshah, kijungs, christos}@cs.cmu.edu.

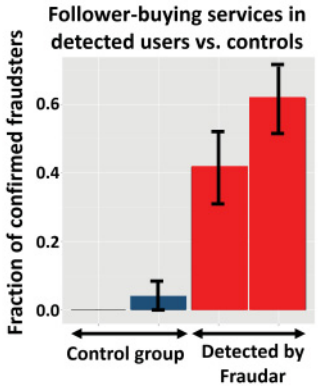
DOI: http://dx.doi.org/10.1145/3056563



(a) Our theoretical bound on detectable region



(b) FRAUDAR outperforms competitors



(c) FRAUDAR detects many confirmed fraudsters

(d) Sample fraudster detected

Fig. 1. (a) *Our theoretical thresholds:* fraudsters in the detection region will be caught by our approach. Our novel optimizations improve the (data-dependent) bounds by lowering it to the green region. (b) FRAUDAR *outperforms competitors.* (c) *A large fraction of accounts flagged by our algorithm are confirmed fraudsters*: both among detected followees (left red bar) and followers (right red bar) compared to almost none among non-flagged accounts (two control groups). Confirmation was done by inspecting Tweets that advertise *TweepMe* or *TweeterGetter*. (d) *Real-life results—a sample fraudster caught.*

misleading consumers about restaurants, hotels, and other services and products. Detecting and neutralizing these actions are important for companies and consumers alike.

The tell-tale sign of such fraudulent actions is that fraudsters must add many edges, creating unusually *large* and *dense* regions in the adjacency matrix of the graph. Smart fraudsters will also try to "look normal," by adding links to popular items/idols (like famous singers/actors or well-liked products)—this behavior is called "*camouflage*" in the recent literature. State-of-the-art algorithms such as SPOKEN [Prakash et al. 2010] and NETPROBE [Pandit et al. 2007] exploit exactly the density signal, but do not account for "camouflage."

We propose FRAUDAR, a novel approach, for successfully detecting fraudsters under camouflage, and we give *provable* limits on undetectable fraud. We provide data-dependent limits on the maximum number of edges a group of fraudulent adversaries can have without being detected, on a wide variety of real-world graphs. As shown in Figure 1(a), FRAUDAR provides limits on undetectable fraud and additionally provides novel optimizations that strengthen this bound.

Moreover, our method outperforms competitors and finds real-world fraud on Twitter. In Figure 1(b), we find that FRAUDAR detects injected fraud with high accuracy, even in the case of camouflage, where prior methods struggle to detect fraudulent attacks. Additionally, when tested on a Twitter graph from 2009, FRAUDAR finds a 4,031 by 4,313 subgraph that is 68% dense. As shown in Figure 1(c) and (d), we find that *a majority* of the detected accounts had tweets showing that they used follower-buying services. Finally, our method is scalable, with near linear runtime in the data size.

Thus, our main contributions are as follows.

—*Metric*: we propose a novel family of metrics that satisfies intuitive "axioms" and has several advantages as a suspiciousness metric.
—*Theoretical Guarantees*: we provide a provable bound on how much fraud an adversary can have in the graph without being caught, even in the face of camouflage. Additionally, we improve the theoretical bound through novel optimizations that better distinguish fraud and normal behavior in real-world data.
—*Effectiveness*: FRAUDAR outperforms state-of-the-art methods in detecting various fraud attacks in real-world graphs and detects a large amount of fraudulent behavior on Twitter.
—*Scalability*: FRAUDAR is scalable, with near-linear time complexity in the number of edges.

Furthermore, FRAUDAR offers natural extensibility and can easily incorporate more complex relations available in certain contexts such as review text, IP addresses, etc.

Reproducibility: Our code is open-sourced at www.andrew.cmu.edu/user/bhooi/camo.zip.

2. BACKGROUND AND RELATED WORK

Fraud detection has received significant focus in recent years. Many existing methods aim to detect fraud through review text [Ott et al. 2011; Jindal and Liu 2008]. However, these approaches are typically not adversarially robust: Spammers can carefully select their review texts to avoid detection. Even without the knowledge of the detection system, they may mimic normal user reviews as closely as possible. Graph-based approaches detect groups of spammers, often by identifying unexpectedly dense regions of the graph of users and products. Such methods are potentially harder to evade, as creating fake reviews unavoidably generates edges in the graph. Graph-based methods may be classified into global and local methods.

Global Methods: Building on singular value decomposition (SVD), latent factor models, and belief propagation (BP), these model the entire graph to find fraud. SpokEn [Prakash et al. 2010] considered the "eigenspokes" pattern produced by pairs of eigenvectors of graphs and was later generalized for fraud detection [Jiang et al. 2014a]. fBox [Shah et al. 2014] builds on SVD but focuses on detecting attacks missed by spectral techniques. Several methods such as CatchSync have used HITS [Kleinberg 1999]-like ideas to detect fraud in graphs [Jiang et al. 2014b; Gyöngyi et al. 2004; Cao et al. 2012; Ghosh et al. 2012; Wu et al. 2006]. BP has been used for fraud classification on eBay [Pandit et al. 2007], and fraud detection [Akoglu et al. 2013]. CrossSpot [Jiang et al. 2015] proposes a likelihood-based metric and a search algorithm for dense block detection in graphs and tensors. All of these methods have been successful in finding fraud but they offer no guarantees of robustness. FBox [Shah et al. 2014] performs adversarial analysis for spectral algorithms, showing that attacks of small enough scale will necessarily evade detection methods that rely on the top k SVD components.

Local Clustering Methods: A different direction for fraud detection focuses on local subgraphs, by analyzing the properties of egonets to detect fraud [Cortes et al. 2001; Perozzi et al. 2014]. CopyCatch [Beutel et al. 2013] and GetTheScoop [Jiang et al. 2014a] use local search heuristics to find relevant dense bipartite subgraphs. However,

without guarantees on the search algorithm, the algorithms may not be robust to intelligent adversaries.

Dense Subgraph Mining: Finding dense subgraphs has been an important focus of graph theory communities and has been studied from a wide array of perspectives [Giatsidis et al. 2011; Karypis and Kumar 1995]. Most closely related to ours is Charikar's work on finding subgraphs with large average degree [Charikar 2000], which shows that subgraph average degree can be optimized with approximation guarantees. Variants have been proposed to efficiently find large, dense subgraphs [Tsourakakis 2015], with approximation guarantees. To our knowledge, however, this is the first work that adapts this theoretical perspective to the challenges of fraud detection and camouflage resistance and achieves meaningful bounds for our application. Moreover, our work differs from these in its setting of bipartite graphs, and in the use of edge re-weighting to further increase accuracy.

Social Network-based Sybil Defense: Multiple identity or "Sybil" attacks pose problems of malicious behavior in distributed systems. SybilGuard [Yu et al. 2006] and SybilLimit [Yu et al. 2008] use a decentralized random walk approach to limit the number of Sybil attackers. SumUp [Tran et al. 2009] and Iolaus [Molavi Kakhki et al. 2013] adapt this to content rating settings. However, these systems rely on a separate trust network between users; our setting is fundamentally different as our approach works directly with the user-product bipartite graph.

Social Spam Analysis in Twitter and other Social Networks: A number of papers analyze the economy behind markets for social spam in social networks [Kanich et al. 2008, 2011] and in Twitter, in particular [Stringhini et al. 2012], collecting data about spamming activity through honeypot [Webb et al. 2008] and other approaches. Broadly in agreement with our findings, they find large markets for spam on Twitter for the purpose of inflating followers and for advertisement (ranging from $20 to $100 for 1000 followers), often making the use of compromised user accounts. Stringhini et al. [2010] use feature-based approaches to detect fraud in social networks; however, this differs from our use of subgraph optimization approaches to detect dense groups of fraudsters.

Robust Collaborative Filtering: Researchers in the recommendation system community have studied the ability of adversaries to distort recommendations [O'Mahony et al. 2004] and developed collaborative filtering algorithms that are hard to manipulate [Mehta et al. 2007; Mehta and Hofmann 2008]. Mehta and Nejdl [2008] and Beutel et al. [2014] directly combine fraud detection with collaborative filtering for more robust recommendations. Researchers have demonstrated that even simple camouflage, such as giving average ratings to most items, deceives common collaborative filtering algorithms [Mobasher et al. 2007]. To combat this type of fraud, researchers assume limits on the knowledge of the adversary. Our approach makes no such assumptions and rather assumes that the adversary has complete knowledge of the graph.

Handling Camouflage: Gu et al. [2015] and Virdhagriswaran and Dakin [2006] consider fraud detection methods that are robust to camouflage attacks. However, both methods focus on the time-series domain, observing changes in the behavior of fraudsters from system access logs rather than graph data.

A comparison between FRAUDAR and other fraud detection algorithms is summarized in Table I. Our proposed method FRAUDAR is the only one that matches all specifications.

3. PROBLEM DEFINITION

Consider a set of m users $\mathcal{U} = \{u_1, \ldots, u_m\}$ and n objects $\mathcal{W} = \{w_1, \ldots, w_n\}$ connected according to a bipartite graph $G = (\mathcal{U} \cup \mathcal{W}, \mathcal{E})$. We can consider the objects to be followees on Twitter or products on Amazon. Table II gives a complete list of the symbols we use throughout the article. We now describe our attack model and then our problem definition.

Table I. Comparison Between FRAUDAR and Other Fraud Detection Algorithms

	[Charikar 2000]	[Tsourakakis 2015]	CopyCatch	CatchSync	BP-based methods	SpokEn	fBox	GetTheScoop	CrossSpot	FRAUDAR
Detects dense blocks	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Camouflage-resistant			✓	?			?			✓
Theoretical guarantees	✓	✓								✓

Table II. Symbols and Definitions

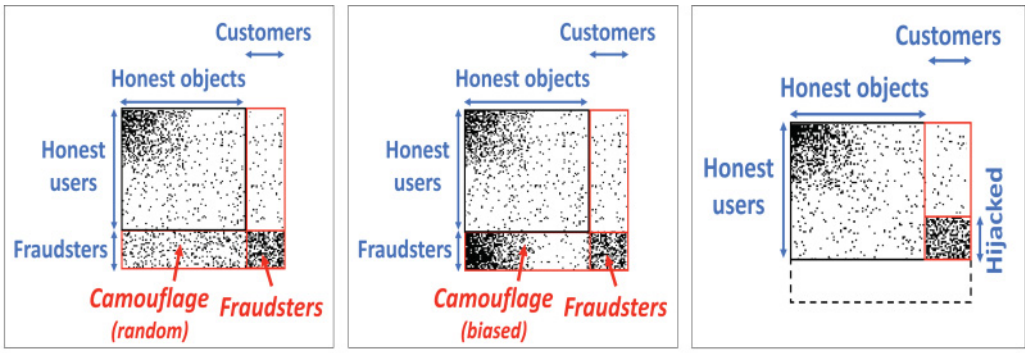
Symbol	Interpretation
$\mathcal{U} = \{u_1, \ldots, u_m\}$	Users
$\mathcal{W} = \{w_1, \ldots, w_n\}$	Objects
$\mathcal{V}$	Nodes of bipartite graph: $\mathcal{U} \cup \mathcal{W}$
G	Bipartite graph $G = (\mathcal{V}, \mathcal{E})$
$\mathcal{A}$	Subset of users
$\mathcal{B}$	Subset of objects
$\mathcal{S}$	Subset of nodes, $\mathcal{S} = \mathcal{A} \cup \mathcal{B}$
$g(\mathcal{S})$	Density metric
$f(\mathcal{S})$	'Total suspiciousness' metric (2)
$\mathcal{X}$	Current set of nodes in the greedy algorithm
Δ_i	$f(\mathcal{X} \setminus \{i\}) - f(\mathcal{X})$
$\hat{\mathcal{A}}, \hat{\mathcal{B}}$	Users (resp. objects) returned by FRAUDAR
m_0, n_0	No. of users (resp. objects) in fraud block
d_i	ith column sum of adjacency matrix
λ	Min. fraction of fraud edges per customer
$g_{\log}$	Logarithmic weighted metric

Attack Model. We assume that fraudsters are hired to use users they control to add edges pointing to a subset of nodes in $\mathcal{W}$. For example, a business may pay for followers on Twitter or positive reviews on Yelp. In general, fraudsters add a large number of edges, inducing a dense subgraph between the fraudster accounts and customers, as shown in the bottom right corner of each subplot of Figure 2. This general characteristic of fraud was found to be true in our experiments on real datasets, as well as in many other papers that use dense blocks to detect fraud [Beutel et al. 2013; Prakash et al. 2010; Jiang et al. 2014a; Pandit et al. 2007; Akoglu et al. 2013].

To mask the fraud, fraudster accounts can add arbitrary "camouflage," i.e., edges pointing from their user accounts to any of the nodes in $\mathcal{W}$ that are not customers. We assume that fraudsters have *complete* knowledge of the graph and fraud detection mechanisms, enabling worst-case camouflage for any fraud detection system we create. Examples of the possible types of camouflage are given in Figure 2: (a) adding camouflage edges to random honest users, (b) camouflage biased toward high degree nodes, and (c) using hijacked accounts, whereby fraudster accounts have realistic patterns of camouflage essentially similar to that of honest users.

The case of hijacked accounts is important in practice—[Stringhini et al. 2012] studies the underground economy of markets for purchasing Twitter followers and finds that attackers often acquire hijacked accounts, both to use them to inflate the follow count of a target account and to promote tweets for advertising purposes.

While it is trivial for fraud accounts to add edges to any other node, it is more difficult for customer accounts to get honest edges. In particular, we assume that a customer



(a) Random camouflage (b) Biased camouflage (c) Hijacked accounts

Fig. 2. *Three examples of possible attacks*: fraudsters attempt to conceal their presence by (a) adding edges toward random honest users, (b) adding edges biased toward popular honest users, or (c) hijacking exist honest user accounts, and using them for fraud.

would try to increase their number of incoming edges by a significant portion, and as a result a fraction, $\lambda \in [0, 1]$, of their incoming edges will be from fraudsters. This assumption would manifest itself as customers wanting to boost their follower count to seem *noticeably* more popular or a restaurant wanting a *significant* number of positive ratings to shift its average "number of stars" on Yelp. We will demonstrate how using this real-world pattern significantly improves fraud detection both theoretically and in practice.

Desired Properties of Detection Approach. Our goal is to detect dense subgraphs in G, typically indicative of fraudulent groups of users and objects, like in the bottom-right of each subplot of Figure 2.

INFORMAL PROBLEM 1. *Given a bipartite graph, detect attacks so as to minimize the number of edges that fraudsters can add pointing to customers without being detected.*

Given that we want our detection algorithm to be able to handle camouflage, we define the requirements for a *camouflage-resistant* algorithm.

Definition 3.1. Let $(\mathcal{A}, \mathcal{B})$ be a block consisting of fraudulent users and objects. A density metric g is *camouflage-resistant* if when any amount of camouflage is added by the adversary, $g(\mathcal{A} \cup \mathcal{B})$ does not decrease.

That is, fraudsters cannot make themselves less suspicious by adding camouflage. Our goal is to find a fraud detection approach satisfying the following criteria.

PROBLEM DEFINITION 1 (DENSE SUBGRAPH DETECTION). *Design a class of density metrics for bipartite graphs, which can be optimized (1) in near-linear time, (2) within a constant factor of the optimum, and (3) is minimally affected by camouflage edges added by adversaries.*

Obtaining theoretical guarantees on the near-optimality of the returned subgraph is important because, as we will later show, it allows us to offer guarantees against *worst-case* fraudsters.

4. PROPOSED METHOD

Given this problem definition and attack model, we now offer FRAUDAR and our theoretical analysis of FRAUDAR.

4.1. Metric

In this section, we propose a class of metrics g that have particularly desirable properties when used as suspiciousness metrics. Namely, we will show that if g takes the form in Equations (1) and (2), then it can be optimized in a way that is (a) scalable, (b) offers theoretical guarantees, and (c) is robust to camouflage.

Let $\mathcal{A} \subseteq \mathcal{U}$ be a subset of users and $\mathcal{B} \subseteq \mathcal{W}$ be a subset of objects. Let $\mathcal{S} = \mathcal{A} \cup \mathcal{B}$, and $\mathcal{V} = \mathcal{U} \cup \mathcal{W}$. For the rest of this article, we use g to denote the density metric that the algorithm will optimize, i.e., the algorithm will find $\mathcal{S}$ to (approximately) maximize $g(\mathcal{S})$. Note that g has a single argument, which is the union of the users and objects whose suspiciousness we are evaluating.

We propose using density metrics g of the following form:

$$g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}, \tag{1}$$

where total suspiciousness f is

$$\begin{aligned} f(\mathcal{S}) &= f_{\mathcal{V}}(\mathcal{S}) + f_{\mathcal{E}}(\mathcal{S}) \\ &= \sum_{i \in \mathcal{S}} a_i + \sum_{i,j \in \mathcal{S} \wedge (i,j) \in \mathcal{E}} c_{ij}, \end{aligned} \tag{2}$$

for some constants $a_i \geq 0$ and constants $c_{ij} > 0$.

Intuitively, the node suspiciousness $f_{\mathcal{V}}(\mathcal{S})$ is a sum of constants a_i corresponding to the users and objects in $\mathcal{S}$, which can be thought of as how individually suspicious that particular user or object is. The edge suspiciousness $f_{\mathcal{E}}(\mathcal{S})$ is a sum of constants c_{ij} corresponding to the edges in between $\mathcal{S}$, which can be thought of as how suspicious that particular edge is (e.g., the suspiciousness of the text of a review by user i for object j).

There are several advantages to metrics of this form. First, metrics of this form can be optimized in a way that is (a) scalable, (b) offers theoretical guarantees, and (c) is robust to camouflage, as we demonstrate in the rest of this article. All three of these properties hold due to the particular chosen form in Equations (1) and (2).

Second, metrics of this form obey a number of basic properties (or axioms) that we would intuitively expect a reasonable suspiciousness metric should meet, as we next show. These basic properties are adapted from the "axioms for suspiciousness metrics," proposed in Jiang et al. [2015], to our setting where node and edge weights exist.

Axiom 1 (Node Suspiciousness). *A subset consisting of higher suspiciousness nodes is more suspicious than one consisting of lower suspiciousness nodes, if the other conditions are fixed. Formally,*

$$|\mathcal{S}| = |\mathcal{S}'| \quad \wedge \quad f_{\mathcal{E}}(\mathcal{S}) = f_{\mathcal{E}}(\mathcal{S}') \quad \wedge \quad f_{\mathcal{V}}(\mathcal{S}) > f_{\mathcal{V}}(\mathcal{S}') \Rightarrow g(\mathcal{S}) > g(\mathcal{S}').$$

Axiom 2 (Edge Suspiciousness). *Adding edges within a subset increases the suspiciousness of the subset if the other conditions are fixed. Formally,*

$$e \notin \mathcal{E} \Rightarrow g(\mathcal{S}(\mathcal{V}, \mathcal{E} \cup \{e\})) > g(\mathcal{S}(\mathcal{V}, \mathcal{E})),$$

where $\mathcal{S}(\mathcal{V}, \mathcal{E})$ is the subgraph induced by $\mathcal{S}$ in the graph $(\mathcal{V}, \mathcal{E})$.

The *edge density* $\rho(\mathcal{S})$ of an induced subgraph is its number of edges divided by its maximum possible number of edges.

Axiom 3 (Size). *Assuming node and edge weights are all equal, larger subsets are more suspicious than smaller subsets with the same edge density. Formally, given $a_i = a \; \forall \; i$, and $c_{ij} = b \; \forall \; (i, j) \in \mathcal{E}$:*

$$|\mathcal{S}| > |\mathcal{S}'| \quad \wedge \quad \mathcal{S} \supset \mathcal{S}' \quad \wedge \quad \rho(\mathcal{S}) = \rho(\mathcal{S}') \Rightarrow g(\mathcal{S}) > g(\mathcal{S}').$$

AXIOM 4 (CONCENTRATION). *A subset with smaller size is more suspicious than one with the same total suspiciousness but larger size. Formally,*

$$|\mathcal{S}| < |\mathcal{S}'| \quad \wedge \quad f(\mathcal{S}) = f(\mathcal{S}') \Rightarrow g(\mathcal{S}) > g(\mathcal{S}').$$

Density metrics g of the form defined in Equation (1) satisfy these properties.

THEOREM 4.1. *The density metric defined in Equation* (1) *satisfies axioms 1 to 4.*

PROOF.
Axiom 1 (Node Suspiciousness).

$$\begin{aligned} g(\mathcal{S}) &= \frac{f_{\mathcal{V}}(\mathcal{S}) + f_{\mathcal{E}}(\mathcal{S})}{|\mathcal{S}|} \\ &> \frac{f_{\mathcal{V}}(\mathcal{S}') + f_{\mathcal{E}}(\mathcal{S}')}{|\mathcal{S}|} = \frac{f_{\mathcal{V}}(\mathcal{S}') + f_{\mathcal{E}}(\mathcal{S}')}{|\mathcal{S}'|} = g(\mathcal{S}'). \end{aligned}$$

Axiom 2 (Edge Suspiciousness). Let $e = (u, v)$.

$$\begin{aligned} g(\mathcal{S}(\mathcal{V}, \mathcal{E} \cup \{e\})) &= \frac{f_{\mathcal{V}}(\mathcal{S}) + f_{\mathcal{E}}(\mathcal{S}) + c_{uv}}{|\mathcal{S}|} \\ &> \frac{f_{\mathcal{V}}(\mathcal{S}) + f_{\mathcal{E}}(\mathcal{S})}{|\mathcal{S}|} = g(\mathcal{S}(\mathcal{V}, \mathcal{E})). \end{aligned}$$

Axiom 3 (Size). Let $\mathcal{S} = \mathcal{A} \cup \mathcal{B}$, and ρ be the edge density.

$$\begin{aligned} g(\mathcal{S}) = \frac{f_{\mathcal{V}}(\mathcal{S}) + f_{\mathcal{E}}(\mathcal{S})}{|\mathcal{S}|} &= a + b\left(\frac{\rho|\mathcal{A}||\mathcal{B}|}{|\mathcal{A}| + |\mathcal{B}|}\right) \\ &= a + b\rho\left(\frac{1}{|\mathcal{A}|} + \frac{1}{|\mathcal{B}|}\right)^{-1}, \end{aligned}$$

which is increasing in both $|\mathcal{A}|$ and $|\mathcal{B}|$.
Axiom 4 (Concentration).

$$g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|} > \frac{f(\mathcal{S})}{|\mathcal{S}'|} = \frac{f(\mathcal{S}')}{|\mathcal{S}'|} = g(\mathcal{S}'). \quad \square$$

Note that a few other simple metrics violate the axioms: the edge density $\rho(\mathcal{S})$ itself, as a metric, violates axiom 3: intuitively, for a fixed density, this metric does not increase as the size of $\mathcal{S}$ increases. On the opposite end, the total edge weight function $f_{\mathcal{E}}(\mathcal{S})$ violates axiom 4 as it does not consider how concentrated the edge weight is. In contrast, average suspiciousness $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$ scales in a reasonable way with $|\mathcal{S}|$ and satisfies both axioms.

A simple example of a metric g as defined in Equations (1) and (2) is the bipartite graph average degree.

Example 4.2. (Bipartite Graph Average Degree) Let $a_i = 0$, and let $c_{ij} = 1$ if $(i, j) \in \mathcal{E}$ and 0 otherwise. In the expression (2) for $f(\mathcal{S})$, we add one term c_{ij} for each edge (i, j) for which i, j are both in the subset $\mathcal{S}$. Thus, $f(\mathcal{S})$ is equal to the number of edges in the subgraph spanned by $\mathcal{S}$, or half the total degree in the subgraph spanned by $\mathcal{S}$. As a result, $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$ is half the average degree of the subgraph spanned by $\mathcal{S}$.

4.2. Interpretation of Size Normalization Approaches

In Section 4.1, we defined our average suspiciousness metric in Equation (1) as $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$. The denominator can be interpreted as a normalization term that cap-

tures our expectation that as $|\mathcal{S}|$ increases, the values of $f(\mathcal{S})$ we expect to observe should increase. In particular, $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$ involves a *linear* normalization term (since its denominator is proportional to $|\mathcal{S}|$), roughly corresponding to an expectation that as $|\mathcal{S}|$ grows, $f(\mathcal{S})$ scales approximately linearly.

We next consider the space of possible choices for the exact form of the normalization term and justify our choice of the linear normalization as in Equation (1). Consider the family of normalizations based on powers of $|\mathcal{S}|$; i.e.,

$$g_\alpha(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|^\alpha}. \tag{3}$$

Examples include the total suspiciousness $g_0(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|^0} = f(\mathcal{S})$, and average suspiciousness $g(\mathcal{S}) = g_1(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$.

The key intuition for $\alpha = 1$, i.e., linear normalization, which we explain in more detail in Appendix A, is that linear normalization roughly corresponds to normalizing by the standard deviation of $f(\mathcal{S})$. Normalizing $f(\mathcal{S})$ by its standard deviation is based on the idea of measuring the suspiciousness of a set of nodes by the number of standard deviations above the mean of $f(\mathcal{S})$. Indeed, for the purpose of anomaly detection, measuring anomalies in units of standard deviations is a common and natural approach (e.g., this is done when using z-scores for anomaly detection). This is because standard deviations are a natural scale measure for the distribution of deviations from the mean that we would typically expect.

4.3. Algorithm

Let f and g be as given in Equations (1) and (2). In this section, we give an algorithm for optimizing the density metric g in near-linear time.

Algorithm 1 describes our proposed FRAUDAR algorithm, a greedy approach inspired by that of Charikar [2000] but which covers our broader objective class. We start with the entire set of nodes $\mathcal{U} \cup \mathcal{W}$, and then repeatedly remove the node that results in the highest value of g evaluated on the remaining set of nodes. Formally, denote by $\mathcal{X}$ the current set we are optimizing over; initially, we set $\mathcal{X} = \mathcal{U} \cup \mathcal{W}$. Let $\Delta_i = f(\mathcal{X} \setminus \{i\}) - f(\mathcal{X})$ be the change in f when we remove i from the current set. At each step, we will select i to maximize Δ_i, i.e., to leave behind the set with highest value of f. We then remove i from $\mathcal{X}$. We then repeat this process: We recompute the values of Δ_j, and then choose the next node to delete, and so on. This leads to a shrinking series of sets $\mathcal{X}$ over time, denoted $\mathcal{X}_0, \ldots, \mathcal{X}_{m+n}$ of sizes $m + n, \ldots, 0$. At the end, we return the one of these that maximizes the density metric g.

The key fact that allows the algorithm to be efficient is the forms for f and g in Equations (1) and (2). When i is removed, the only values of Δ_j that need to be updated are those where j is a neighbor of i. This is because for all other j, the expressions (1) and (2) ensure that Δ_j does not change. Hence, the updates are fast: for each $(i, j) \in \mathcal{E}$, over the lifetime of the algorithm we will perform at most one such update over this edge, for a total of $O(|\mathcal{E}|)$ updates. Using appropriate data structures, as we next describe, each update can be performed in $O(\log |\mathcal{V}|)$ time, totalling $O(|\mathcal{E}| \log |\mathcal{V}|)$ time.

Priority Tree. Each element $i \in \mathcal{X}$ has a priority that will change as the algorithm progresses: The priority of element i at the tth iteration is $\Delta_i = f(\mathcal{X}_t \setminus \{i\}) - f(\mathcal{X}_t)$. This ensures that in Line 5, the element $i^* = \arg\max_{i \in \mathcal{X}_t} g(\mathcal{X}_t \setminus \{i\})$ we wish to find is exactly the element of highest priority, allowing us to retrieve it quickly (in $O(\log |\mathcal{V}|)$ time, as we explain below). Note that it does not matter if we use f or g in the arg max since the denominator of g in Equation (1), $|\mathcal{X}_t \setminus \{i\}|$, is the same for all possible deletions i.

ALGORITHM 1: FRAUDAR which greedily removes nodes to maximize a metric g. Line 5 and 6 run in $O(\log|\mathcal{V}|)$ time, using a data structure described in Section 4.3.

Require: Bipartite $G = (\mathcal{U} \cup \mathcal{W}, \mathcal{E})$; density metric g of the form in Equation (1)
1: **procedure** FRAUDAR(G, g)
2: Construct priority tree T from $\mathcal{U} \cup \mathcal{W}$ ▷ see Section 4.3
3: $\mathcal{X}_0 \leftarrow \mathcal{U} \cup \mathcal{W}$ ▷ suspicious set is initially the entire set of nodes $\mathcal{U} \cup \mathcal{W}$
4: **for** $t = 1, \ldots, (m+n)$ **do**
5: $i^* \leftarrow \arg\max_{i \in \mathcal{X}_i} g(\mathcal{X}_i \setminus \{i\})$ ▷ exonerate least suspicious node
6: Update priorities in T for all neighbors of i^*
7: $\mathcal{X}_t \leftarrow \mathcal{X}_{t-1} \setminus \{i^*\}$
8: **end for**
9: **return** $\arg\max_{\mathcal{X}_i \in \{\mathcal{X}_0, \ldots, \mathcal{X}_{m+n}\}} g(\mathcal{X}_i)$ ▷ return most suspicious set $\mathcal{X}_i$
10: **end procedure**

These priorities are stored in the priority tree T constructed in line 2 of Algorithm 1. This data structure is a binary tree with all $|\mathcal{V}|$ elements as leaves, all at the bottom level of the tree. Each internal node keeps the track of the maximum priority of its two children.

The priority tree supports fast retrieval of the maximum priority element (used in Line 5 of Algorithm 1); it does this by starting at the root and repeatedly moving to the child with higher priority. It also supports quickly updating priorities: Since all the leaves can be stored in fixed locations, we can easily retrieve the leaf at any index to update its priority. Then, after updating that node's priority, we travel up the tree to update each parent up to the root (used in Line 6). Each of these operations on T takes $O(\log|\mathcal{V}|)$ time.

Scalability. The bottleneck is the loop in Lines 5 to 7 that runs $m+n$ times. Lines 5 and 6 take $O(\log|\mathcal{V}|)$ as discussed, while Line 7 is constant time. Finally, we need $|\mathcal{E}|$ updates to node priorities, one for each edge. Thus, the algorithm takes $O(|\mathcal{E}|\log|\mathcal{V}|)$ time. Its space complexity is $O(|\mathcal{E}|)$ if we store the whole graph in memory. However, if we store the graph externally and only retrieve neighbor lists of individual nodes when they are needed (i.e., before the node is deleted, in Line 6 of Algorithm 1), the space complexity reduces to $O(|\mathcal{V}|)$, which is the space needed to store the priority trees, since these trees contain at most $O(|\mathcal{V}|)$ nodes.

4.4. Theoretical Bounds

So far, we have shown that g can be optimized in near-linear time. In this section, we will show that when f and g are of the form (1) and (2), FRAUDAR is guaranteed to return a solution of at least half of the optimum value.

Note that the metric g, treated as a set function, is in general neither submodular (i.e., $\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{V}, g(\mathcal{A}) + g(\mathcal{B}) \geq g(\mathcal{A} \cup \mathcal{B}) + g(\mathcal{A} \cap \mathcal{B})$) or supermodular (i.e., $\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{V}, g(\mathcal{A}) + g(\mathcal{B}) \leq g(\mathcal{A} \cup \mathcal{B}) + g(\mathcal{A} \cap \mathcal{B})$). For simplicity, we will set all node weights a_i to 0 and all edge weights c_{ij} to 1. To show that g is not submodular, consider a graph with only two nodes, i and j, joined by an edge; then

$$g(\{i\}) + g(\{j\}) = 0 < g(\{i, j\}) + g(\emptyset).$$

For supermodularity, consider a graph with three nodes, i, j, k, and edges ik and jk. Then,

$$g(\{i, k\}) + g(\{j, k\}) = 1/2 + 1/2 > 2/3 + 0 = g(\{i, j, k\}) + g(\{k\}).$$

Since g is not submodular or supermodular, we cannot use the standard efficient approximate or exact algorithms for these (such as the $1 - 1/e$ approximation factor

greedy algorithm [Nemhauser et al. 1978]). However, a different approach used by Charikar [2000] does allow us to obtain a 1/2 approximation guarantee.

THEOREM 4.3. *Let $\mathcal{A}, \mathcal{B}$ be the set of users and objects returned by* FRAUDAR. *Then,*

$$g(\mathcal{A} \cup \mathcal{B}) \geq \frac{1}{2} g_{OPT},$$

where g_{OPT} is the maximum value of g, i.e.,

$$g_{OPT} = \max_{\mathcal{A}', \mathcal{B}'} g(\mathcal{A}' \cup \mathcal{B}').$$

PROOF. Let the elements of $\mathcal{V}$ be labeled $v_1, v_2, \ldots, v_{m+n}$. We define "weight assigned to node v_i in $\mathcal{S}$" as

$$w_i(\mathcal{S}) = a_i + \sum_{(v_j \in \mathcal{S}) \wedge ((v_i, v_j) \in \mathcal{E})} c_{ij} + \sum_{(v_j \in \mathcal{S}) \wedge ((v_j, v_i) \in \mathcal{E})} c_{ji},$$

where $a_i (\geq 0)$ indicates the weight of node v_i and $c_{ij} (> 0)$ indicates that of edge (v_i, v_j) as in Equation (2). Note that when node v_i is removed from the current set $\mathcal{S}$ at some point in the algorithm, $w_i(\mathcal{S})$ is the decrease in the value of f, since it is the sum of all terms excluded in Equation (2) when node v_i is removed.

Now consider the optimal set $\mathcal{S}^*$. For each node $v_i \in \mathcal{S}^*$, we claim that $w_i(\mathcal{S}^*) \geq g(\mathcal{S}^*)$. Otherwise, removing a node with $w_i(\mathcal{S}^*) < g(\mathcal{S}^*)$ results in

$$\begin{aligned} g' &= \frac{f(\mathcal{S}^*) - w_i(\mathcal{S}^*)}{|\mathcal{S}^*| - 1} > \frac{f(\mathcal{S}^*) - g(\mathcal{S}^*)}{|\mathcal{S}^*| - 1} \\ &= \frac{f(\mathcal{S}^*) - f(\mathcal{S}^*)/|\mathcal{S}^*|}{|\mathcal{S}^*| - 1} = g(\mathcal{S}^*), \end{aligned}$$

which is a contradiction.

Let v_i be the node that FRAUDAR removes first among those in $\mathcal{S}^*$, and let $\mathcal{S}'$ be the set before FRAUDAR removes v_i. Then, since $\mathcal{S}' \supset \mathcal{S}^*$, $w_i(\mathcal{S}') \geq w_i(\mathcal{S}^*)$. Moreover, since FRAUDAR chooses to remove node v_i, for each of the other remaining nodes $v_j \in \mathcal{S}'$, $w_j(\mathcal{S}') \geq w_i(\mathcal{S}')$. Summing over j (ranging over $v_j \in \mathcal{S}'$), the left side consists of the summands of $f(\mathcal{S}')$, each appearing at most twice since each edge score is assigned to two nodes (namely, the two nodes incident to it), and each node score is assigned to one node. Thus, the sum is at most $2f(\mathcal{S}')$, while the right side sums to $|\mathcal{S}'| w_i(\mathcal{S}')$. Hence, summing over j gives $f(\mathcal{S}') \geq \frac{|\mathcal{S}'| w_i(\mathcal{S}')}{2}$. Also, note that $g(\mathcal{A} \cup \mathcal{B}) \geq g(\mathcal{S}')$ since $\mathcal{A} \cup \mathcal{B}$ is the best set that FRAUDAR encountered, and $\mathcal{S}'$ is one of the sets that it encountered. We conclude that

$$g(\mathcal{A} \cup \mathcal{B}) \geq g(\mathcal{S}') = \frac{f(\mathcal{S}')}{|\mathcal{S}'|} \geq \frac{w_i(\mathcal{S}')}{2} \geq \frac{w_i(\mathcal{S}^*)}{2} \geq \frac{g(\mathcal{S}^*)}{2}. \quad \square$$

4.5. Edge Weights and Camouflage Resistance

So far, we have seen that metrics of the form: $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$, where $f(\mathcal{S}) = \sum_{i \in \mathcal{S}} a_i + \sum_{i,j \in \mathcal{S} \wedge (i,j) \in \mathcal{E}} c_{ij}$ can be optimized efficiently and with approximation guarantees. In this section, we show how we can select metrics within this class that are resistant to camouflage, i.e., they do not allow fraudulent users to make themselves less suspicious by adding *camouflage edges*, i.e., edges toward honest objects.

Recall that a_i and c_{ij} are the weights of node i and edge ij, while $f(\mathcal{S})$ is the total node and edge weight in $\mathcal{S}$, respectively. A key idea of our approach is that instead of treating every edge equally, we assign a lower weight c_{ij} when the target object j has

high degree. This is because objects of very high degree are not necessarily suspicious: For example, on Facebook, popular pages may be expected to gather extremely large numbers of "likes." Indeed, Bhamidi et al. [2015] found that in certain large Twitter graphs, the largest degree nodes are connected to a constant fraction of the number of nodes in the graph, in some cases exceeding half the total number of nodes. Without downweighting based on node degree, the densest subgraphs are likely to be composed of a group of extremely high degree nodes, rather than nodes that are suspicious. Hence, downweighting allows us to put greater emphasis on groups of users and objects that are *unexpectedly* dense, in the sense of having nodes of relatively low degree but where an extremely large fraction of these edges lie within a dense subgraph.

Downweighting high degree columns also incorporates the intuitive assumption that fraudulent objects are expected to have a significant fraction of edges from fraud; otherwise, they would not be benefiting appreciably from the fraud. Thus, an object with 100 edges, all within the dense subblock, is more likely to be fraudulent than an object with 1,000 edges, 100 of them within the dense subblock, and the former object's edges should be given higher weights as a result.

Hence, if we consider the adjacency matrix with rows representing users and columns representing objects, we would like to downweight columns with high column sum (column-weighting). A simple result we show in this section is that column-weightings are camouflage resistant. Recall that a density metric g is camouflage-resistant if $g(\mathcal{A} \cup \mathcal{B})$ does not decrease when any amount of camouflage is added by an adversary with fraudulent users $\mathcal{A}$ and customers $\mathcal{B}$. Let d_i be the ith column sum, i.e., the degree of object i.

Formally, define a *column-weighting* as a choice of weighting in which each c_{ij} is a function of the respective column sum, i.e., $c_{ij} = h(d_j)$ for some function h.

THEOREM 4.4. *Let c_{ij} be a column-weighting. Then, g (as defined in Equations* (1) *and* (2)*) is camouflage resistant.*

PROOF. Adding camouflage only adds edges in the region between $\mathcal{A}$ (fraudulent users) and $\mathcal{B}^C$ (honest objects). It does not add or remove edges within the fraudulent block; moreover, the weights of these edges do not change either as their weights only depend on the column degrees of $\mathcal{B}$, which do not change when camouflage is added. Thus, the value of g does not change. □

A natural follow-up question is whether camouflage resistance also holds for row-weightings (i.e., selecting c_{ij} to be a function of the corresponding row sum). It turns out that row-weightings are, in general, not camouflage resistant. This is because a fraudulent user account can add a large number of camouflage edges, thereby increasing their row sum, decreasing the weight of each of their edges. Thus, $g(\mathcal{A} \cup \mathcal{B})$ decreases, meaning that g is not camouflage resistant.

Hence, we may choose any column-weighting, while ensuring camouflage resistance. The remaining question is what function to choose for the column-weighting, i.e., the function h where $c_{ij} = h(d_j)$. It should be decreasing (so as to downweight columns with high sum). It should shrink more slowly than $h(x) = 1/x$, since $h(x) = 1/x$ allows a single edge to contribute as much as the total contribution of a column with any number of edges, causing us to catch columns with single ones rather than dense blocks.

Within the remaining space of choices, we note that a very similar problem of downweighting based on column frequency appears in deciding the form of the "inverse document frequency" term of the popular heuristic *tf-idf* weighting scheme [Rajaraman et al. 2012], in which logarithmic weighting of frequency has been empirically found to perform well. We also show empirical results (in Section 5.1) that logarithmic weighting leads to strong theoretical bounds. For these reasons, we recom-

mend using $h(x) = 1/\log(x+c)$, where c is a small constant (set to 5 in our experiments) to prevent the denominator from becoming zero, or excessive variability for small values of x. We use the resulting density metric (denoted $g_{\log}$) in our experiments.

4.6. Implications: Bounding Fraud

Figure 1(a) shows curves representing our theoretical bounds on the maximum amount of fraud that can be present for each possible size of the fraudulent block, based on Theorem 4.3. We now explain how such bounds can be computed from Theorem 4.3. Assume that the fraudulent block contains m_0 user accounts and n_0 customers.

In this section, we assume that no side information is present, so we set a_i, the prior suspiciousness of each node, to 0. Thus, here $g_{\log}(\mathcal{S}) = \frac{1}{|\mathcal{S}|}\sum_{i,j\in\mathcal{S}}\frac{1}{\log(d_j+c)}$, where d_j is the degree of the jth object. Consider a fraudulent subgraph with m_0 user nodes and n_0 object nodes. Assume that each fraudulent customer has at least a certain fraction $0 < \lambda < 1$ of fraudulent edges: Each fraudulent customer should be receiving at least a comparable fraction of fraudulent reviews to its actual honest reviews; otherwise it would not be profiting appreciably from the fraud.

THEOREM 4.5. *Let* $(\hat{\mathcal{A}}, \hat{\mathcal{B}})$ *be the block detected by* FRAUDAR. *Then, the number of edges that a fraudulent block of size* (m_0, n_0) *can have without being detected is at most* $2(m_0 + n_0)g_{\log}(\hat{\mathcal{A}} \cup \hat{\mathcal{B}})\log(m_0/\lambda + c)$. *In other words, our algorithm will detect a fraudulent block without fail if it contains more edges than this threshold.*

PROOF. By Theorem 4.3, $2g_{\log}(\hat{\mathcal{A}} \cup \hat{\mathcal{B}})$ is an upper bound on the value of $g_{\log}$ on any subgraph of users and objects. Since the fraudulent block has $m_0 + n_0$ nodes in total; thus, $2(m_0 + n_0)g_{\log}(\hat{\mathcal{A}} \cup \hat{\mathcal{B}})$ is an upper bound on the value of total suspiciousness $f_{\log}$.

Moreover, each fraudulent customer has at most m_0 fraudulent edges joined to it, and since at least λ fraction of its edges must be fraudulent, it can have at most m_0/λ degree in total. Hence, the weight of each fraudulent edge is at least $\frac{1}{\log(m_0/\lambda+c)}$. But since the total weighted degree is at most $2(m_0 + n_0)g_{\log}(\hat{\mathcal{A}} \cup \hat{\mathcal{B}})$, it follows that the number of fraudulent edges is at most $2(m_0 + n_0)g_{\log}(\hat{\mathcal{A}} \cup \hat{\mathcal{B}})\log(m_0/\lambda + c)$. □

We apply this bound to real data in Section 5.1.

In Appendix B, we give (1) necessary conditions and (2) sufficient conditions for FRAUDAR to detect fraudsters perfectly (i.e., with 100% precision and recall), as well as (3) necessary conditions and (4) sufficient conditions for FRAUDAR to detect at least one fraudulent node correctly.

From a high level, these conditions are based on the *coreness* of nodes, a property of nodes that we define in Appendix B; the necessary and sufficient conditions then involve comparisons between the coreness of the honest nodes and the fraudulent nodes.

5. EXPERIMENTS

We design experiments to answer the following questions.

Q1. Illustration of our theorem: How strong are the bounds that FRAUDAR provides in terms of bounding undetectable fraud in the graph? Does column weighting improve those bounds?

Q2. Evaluation on synthetic data: How accurately does FRAUDAR detect injected fraud under different types of camouflage attacks? Does FRAUDAR outperform state-of-the-art competitors?

Q3. Effectiveness in real-world data: Does FRAUDAR detect true fraud in real-world graphs? Have the fraudulent accounts already been detected by previous methods?

Q4. Scalability: Is FRAUDAR scalable with regard to the data size?

Table III. Bipartite Graph Datasets used in Our Experiments

	# of nodes	# of edges	Density	Content
Amazon [McAuley and Leskovec 2013]	28K (24K,4K)	28K	2.7e-4	Review
Trip Advisor [Wang et al. 2011]	84K (82K,2K)	90K	5.9e-4	Review
Epinion [Leskovec et al. 2010]	264K (132K,132K)	841K	4.8e-5	Who-trust-whom
Wiki-vote [Leskovec et al. 2010]	16K (8K,8K)	103K	1.5e-3	Vote

We implemented FRAUDAR in Python; all experiments were carried out on a 2.4GHz Intel Core i5 Macbook Pro, 16GB RAM, running OS X 10.9.5. The code is available for download at www.andrew.cmu.edu/user/bhooi/camo.zip. We test FRAUDAR on a variety of real-world datasets. Table III offers details on the datasets we used.

To test the accuracy of our method, we use synthetic attacks injected into our Amazon dataset. We structure our "attacks" as shown in Figure 2. We injected a fraudulent block of users and customers with varying densities.

5.1. Q1. Illustration of Our Theorem

In Figure 1(a), we showed our theoretical bounds (Theorem 4.5) applied to compute the maximum number of edges an adversary with $m_0 = 50$ user nodes and $\lambda = 0.5$ can have for various values of n_0, as an illustration of our theoretical bounds. The curves in Figure 1(a) are computed by running FRAUDAR under two weighting schemes. First, we use our $g_{\log}$ scheme exactly as in Theorem 4.5 to get an upper bound $2(m_0 + n_0)g_{\log}(\hat{\mathcal{A}} \cup \hat{\mathcal{B}}) \log(m_0/\lambda + c)$ on the number of fraudulent edges; plotting this against n_0 gives the green region (improved) in Figure 1(a). The blue region (original) comes from using the analogous procedure without the log-weighting, i.e., where $g(\mathcal{S})$ is half the average degree, as in Example 4.2.

In this case, we see that the log-weighted scheme provides stronger bounds, since the bound is lower, i.e., an adversary should have fewer edges in order not to be detected. Intuitively, this happens because down-weighting high degree columns decreases the weight of many of the honest high degree objects in the dataset, so groups of adversaries stand out more, resulting in stronger bounds on how many edges an adversary can have.

Next, we apply our FRAUDAR in the same way over various real-world graphs to analyze the theoretical upper bounds computed by FRAUDAR on the density that fraudulent blocks can have. We run FRAUDAR on four real-world graphs: *Amazon* [McAuley and Leskovec 2013], *Trip Advisor* [Wang et al. 2011], *Epinions* [Leskovec et al. 2010], and *Wiki-vote* [Leskovec et al. 2010]. The detailed description of each graph is in Table III. For all datasets, Figure 3 shows the maximum number of fraudulent edges that an adversary can have without being detected, assuming 50 fraudulent users and varying the number of fraudulent customers. We see that we can detect fraud most easily in *Trip Advisor*, followed by *Epinion, Wiki-vote, Amazon*; even a fairly sparse block of density around 0.05 would stand out strongly in the *Trip Advisor* graph. While density is important in determining how easy it is to detect fraud in each graph (fraudulent blocks stand out more strongly in a sparse graph), it is not the only factor. Indeed, *Wiki-vote* is actually denser than *Amazon*. In fact, the difficulty of detecting fraud in each graph is mainly determined by its densest blocks, since an adversarial block that is significantly less dense than the densest normal blocks in the graph is unlikely to be detected.

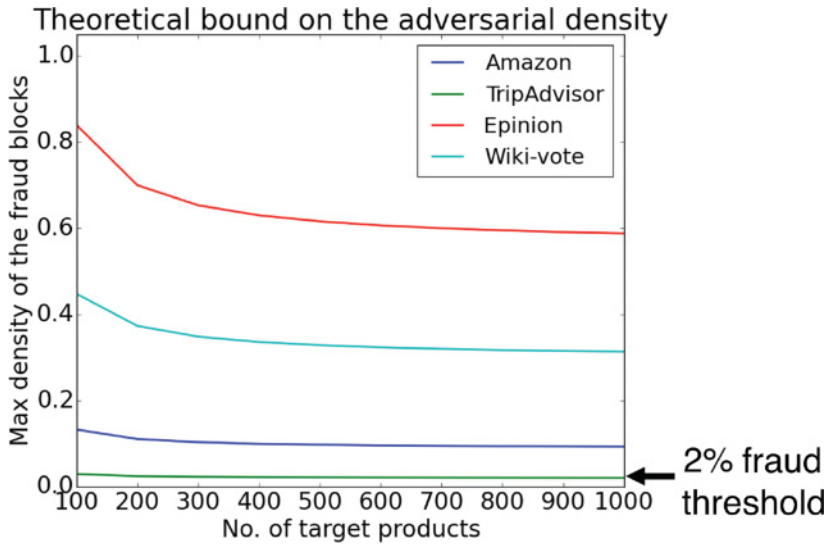


Fig. 3. FRAUDAR *'s bounds on fraud are stringent, on real graphs:* For example, on TripAdvisor, the bound says that a fraudulent block containing 50 user accounts and anywhere between 100 and 1,000 products must have density of <2% to avoid detection.

5.2. Q2. Evaluation on Synthetic Data

In Figure 1(b), we demonstrated that FRAUDAR can effectively detect fraud under following four types of camouflage attacks: (1) injection of fraud with no camouflage, (2) random camouflage, (3) biased camouflage, and (4) hijacked accounts, more accurately than competitors.

We conduct experiments based on the settings at the beginning of this section, averaged over five trials. For the camouflage scenarios (2) and (3), the amount of camouflage added per fraudulent user account was (on average) equal to the amount of actual fraudulent edges for that user. For the "Random Camo" case, for each fake user node, camouflage edges were chosen at random, with on average the same number of camouflage edges as fraudulent edges, as shown in Figure 2(a). For the "Biased Camo" case, for each fake user node, camouflage edges were directed toward each object with probability proportional to the degree of the object as shown in Figure 2(b). For the "Hijacked" case, we used a random subset of existing users to form the fraudulent block.

In each case, we injected 200 fraudulent users and 200 fraudulent products with various edge densities to the subsetted Amazon review graph of 2,000 users and 2,000 products, with a density of 0.0006. We compare FRAUDAR to SPOKEN in their F measure ($= \frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}}$) in detecting the fake users. In the first set of experiments, we assume that no honest user added an edge to the fraudulent target (i.e., object) nodes.

As seen in Figure 1(b), the results demonstrate that FRAUDAR works robustly and efficiently against all four attacks, achieving F-measures of over 0.95 on all four scenarios for densities of at least 0.04. In particular, FRAUDAR is equally accurate in the non-camouflaged, random camouflage, and hijacked settings; this is not surprising as the design of the metric g makes it ignore the effect of camouflage in most settings. A somewhat surprising finding is that biased camouflage ends up being easier to detect than the other cases in our experiment, particularly when the density of the injected subgraph is low. This turns out to be because the camouflage ends up being concentrated

between the fraudulent nodes and a small set of extremely popular products, resulting in this combined set of nodes being caught by the algorithm. Since the set of popular products being caught is small, the resulting accuracy ends up being higher than when camouflage is absent, when the fraudsters may be missed entirely.

On the other hand, SpokEn was able to reach its maximum performance of 0.9 only when fraud blocks had densities of higher than 0.06 and under the "no camouflage" scenario.

The experimental results in Figure 1(b) were based on the assumption that no honest user added an edge to the fraudulent target nodes. However, in a real-world environment, some honest users may add edges to the fraudulent target nodes (which we refer to as "reverse camouflage"). To incorporate this, we conducted another experiment using an attack model where we add edges between honest users and the fraudulent target nodes, but with sparser density compared to the fraud blocks. We added random edges to this region, with half the density of the fraud blocks. All other experimental settings were unchanged. The experimental results are shown in Figure 4(a). For FRAUDAR, the results are generally similar. In contrast, SpokEn shows slightly worse performance under this additional camouflage.

To show that FRAUDAR is effective both at catching fraudulent users accounts as well as fraudulent objects, we next separately evaluate the fraud detection of both fake users and fake targets using F measure. The basic experimental setup is the same as before, with the density of the fraudulent blocks now fixed to 0.03. In Figure 4(b), the bar plots are shown for the comparison. "User wise" (red) denotes the F measure of the detecting fake users, and "target wise" denotes the F measure of detecting fake target nodes. We see that, in general, accuracy is high and fairly similar, but the performance in detecting fake users is slightly higher than that of detecting products.

5.3. Q3. Effectiveness on Real Data

In this section, we verify that FRAUDAR accurately detects a large block of fraudulent accounts in the Twitter follower–followee graph, as verified by hand labeling and by clear signs of fraud exhibited by a majority of the detected users. Indeed, a majority of the detected accounts had tweets advertising follower-buying services, and the tweets had not been removed or the accounts suspended for the 7 years since the data were collected. Figure 1(d) shows a sample fraudster caught by FRAUDAR.

The Twitter graph we use contains 41.7 million users and 1.47 billion follows; it was extracted in July 2009 and first used in Kwak et al. [2010]. On this graph, FRAUDAR detected a dense subgraph of size 4,031 followers by 4,313 followees. This subgraph is extremely dense, with 68% density, which is highly suspicious in itself.

To further investigate this block, we randomly sampled 125 followers and 125 followees in the block detected by FRAUDAR for hand labeling to determine how many of them appear fraudulent. To do this, we labeled which users were fraudulent based on the following characteristics of their profile data, chosen based on established criteria in the literature [Shah et al. 2014] summarized below.

—links on profile associated with malware or scams
—clear bot-like behavior (e.g., replying to large numbers of tweets with identical messages)
—account deleted
—account suspended

For comparison, we also construct two control groups of size 100 containing users that were not detected by the algorithm. The first control group contains randomly selected non-detected users. For the second (degree-matched) control group, we constructed it to match the follower count of users in the detected group; we do this by repeatedly

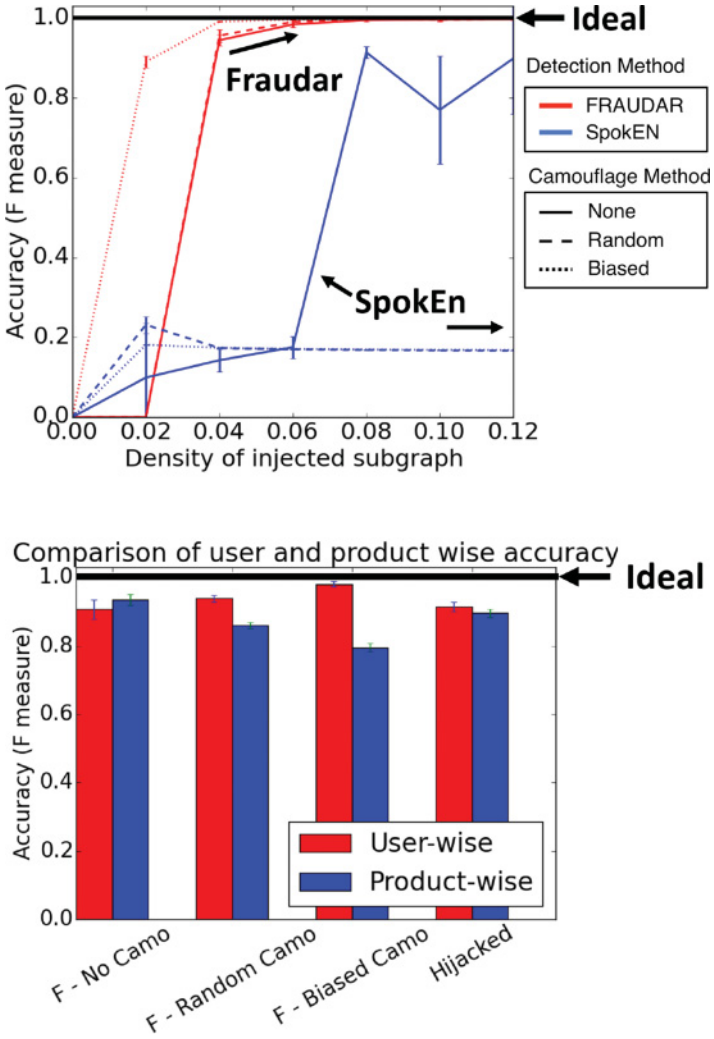


Fig. 4. (a) FRAUDAR *outperforms competitors in multiple settings*. Accuracy of fraud detection on Amazon data in the experiment with "reverse camouflage" (edges from honest users to fraudulent products). (b) FRAUDAR *has similar and high accuracy both in detecting fraudulent users and fraudulent customers*. Comparison of accuracy on fake users and targets under four different camouflage attacks.

selecting a random detected user, and then finding another non-detected user who has at most 10% bigger or smaller follower count. During the labeling process, we shuffled the detected users with the control groups randomly and hid group memberships from labelers, labeling users in a "blind" manner.

Additionally, we also check and report how many of these users have Tweets containing the URLs of two known follower-buying services, *TweepMe* and *TweeterGetter*, showing that they had advertized these follower-buying services through tweets.

Note that this entire labeling process used only profile and tweet data and not follower–followee data, whereas our algorithm uses only follower–followee data, so the labeling is a fair estimate of the algorithm's accuracy. We present two pieces of evidence that strongly indicates fraud in the detected group. The strongest evidence concerns the

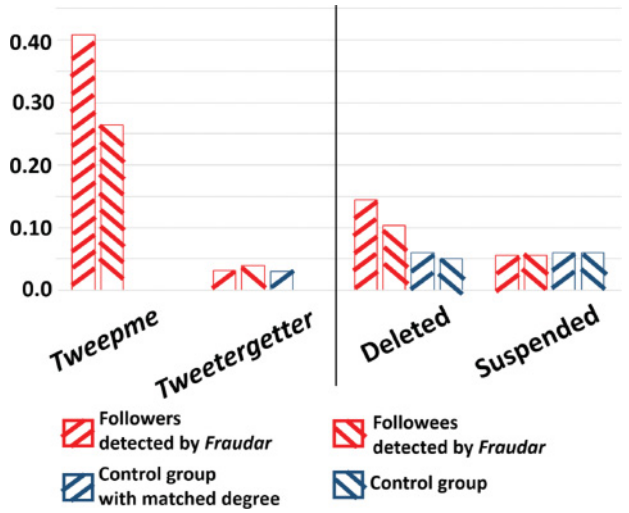


Fig. 5. FRAUDAR *detects a large, clearly fraudulent block in Twitter.* The bars show the fraction of detected and control group accounts who were found to use known follower-buying services, *TweepMe* and *TweeterGetter*, or were deleted or suspended. More than half the detected followers fell into at least one of these categories. In comparison, the control groups had much less detected fraud.

Fig. 6. *Follower-buying services:* a large fraction of detected accounts use *TweepMe* (bottom) or *TweeterGetter* (middle, top). Both are follower-buying services, as can be seen from the banners.

extremely large percentage of users with tweets advertising *TweepMe* or *Tweetergetter*, two known follower-buying services (the banners in Figure 6 show clearly that these are follower-buying services). As shown in Figure 1(c), among the group of followers in the block detected by FRAUDAR, 62% of users had tweets advertizing these two services; this number was 42% among the followees detected by FRAUDAR.

In contrast, in the control groups, there were no mentions of *TweepMe* and very few mentions of *TweeterGetter*, as shown in Figure 5. Figure 5 shows the breakdown of our detected (red) and control (blue) groups in terms of deleted and suspended users, and how many users had tweets advertising *TweepMe* and *TweeterGetter*. Note that the slightly lower percentages of *TweepMe* and *TweeterGetter* in this figure (41% of the detected followers, and 26% for the detected followees) are because Figure 1 ignores

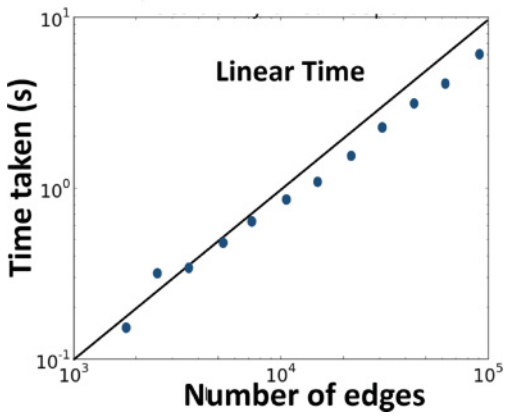


Fig. 7. FRAUDAR *runs in near-linear time*: the curve *(blue)* shows the running time of FRAUDAR, compared to a linear function *(black)*.

deleted, protected, and suspended accounts, for which profile and tweet information were unavailable, making it impossible to determine whether follower-buying services were present. Given the sparsity of *TweepMe* and *TweeterGetter* in the control groups, we see that the detected users are likely characterized by a large block of users using these and possibly other follower-buying services, resulting in a dense block.

Second, we used our hand labeling using the above criteria to determine how many of each group appear either fraudulent or consist of suspended or deleted accounts. 57% of the detected followers and 40% of the followees were labeled as fraudulent, deleted, or suspended accounts, but much fewer in the control groups, with 25% for the degree-matched control group, and 12% for control group with no condition. Thus, both these results support the effectiveness of FRAUDAR in detecting fraudulent users in the real-world graphs.

5.4. Q4. Scalability

Figure 7 shows the near-linear scaling of FRAUDAR's running time in the number of edges. Here, we used the Trip Advisor dataset, and subsampled user nodes in proportions of $0.7^0, \ldots, 0.7^{12}$. Slopes parallel to the main diagonal indicate linear growth.

6. CONCLUSION

In this article, we propose FRAUDAR, a fraud detection algorithm that provably bounds the amount of fraud adversaries can have, even in the face of camouflage. This problem has applications to fraud detection in a wide variety of social network settings, particularly in adversarial settings, such as detecting purchased follows on Twitter, fake likes on Facebook, and rating manipulation on rating platforms. Our main contributions are as follows.

—*Metric*: we propose a novel family of metrics that satisfies intuitive "axioms" and has several advantages as a suspiciousness metric.
—*Theoretical Guarantees*: we provide theorems (see Theorem 4.3 in Section 4.4 and Theorem 4.5 in Section 4.6) on how FRAUDAR gives a provable upper bound on undetectable fraud. We also prove that our proposed metric is camouflage-resistant.
—*Effectiveness*: FRAUDAR was successfully applied on real-world graphs on fraud attacks with various types of camouflage, and outperformed the competitor. It also detected a large block of fraudulent activity in the Twitter follower–followee graph.
—*Scalability*: FRAUDAR runs near-linearly in the input size (see Figure 7).

However, we believe there are still many directions for possible extension of this work. Among others, an interesting problem is how to incorporate temporal information or other additional features, possibly based on the intuition for fraudulent clusters tend to appear in short bursts; this would greatly improve the algorithm's ability to distinguish between honest but high-degree clusters and truly fraudulent clusters. An interesting theoretical direction would be to more carefully characterize the space of objective functions that can be efficiently approximated, as well as better understanding when this can be done with theoretical guarantees, possibly expanding the space of metrics g that we can make use of.

APPENDIXES

A. INTERPRETATION OF SIZE NORMALIZATION APPROACHES

In this appendix, we justify the choice of normalization approach in the metric $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$, from the perspective that it approximately normalizes $f(\mathcal{S})$ by its standard deviation, assuming a simplified setting.

For simplicity, set node weights to zero and all edge weights $c_{ij} = 1$. Then, $f(\mathcal{S})$ is the number of edges in the induced subgraph of $\mathcal{S}$. Further assume an Erdos Renyi like model where each possible edge in the bipartite graph is independently present with probability p. Recalling that $\mathcal{S} = \mathcal{A} \cup \mathcal{B}$, this induced subgraph has $|\mathcal{A}| \cdot |\mathcal{B}|$ possible edges, each present with probability p. Thus, $f(\mathcal{S})$ follows a binomial distribution $\text{Binom}(|\mathcal{A}| \cdot |\mathcal{B}|, p)$, which has standard deviation $\sqrt{|\mathcal{A}| \cdot |\mathcal{B}| \cdot p(1-p)}$. Assuming $|\mathcal{A}|$ and $|\mathcal{B}|$ are both roughly linear in $|\mathcal{S}|$ (which occurs if the sizes $|\mathcal{A}|$ and $|\mathcal{B}|$ and close to one another), this standard deviation is linear in $|\mathcal{S}|$ as well. Summarizing our discussion so far, as $|\mathcal{S}|$ grows, the standard deviation of $f(\mathcal{S})$ scales approximately linearly in $|\mathcal{S}|$, so that normalizing by $|\mathcal{S}|$ can be interpreted as normalizing by the standard deviation of $f(\mathcal{S})$.

Since the standard deviation we computed is also proportional to $\sqrt{|\mathcal{A}| \cdot |\mathcal{B}|}$, this suggests $\sqrt{|\mathcal{A}| \cdot |\mathcal{B}|}$ as a possible alternative normalization scheme, i.e., using the metric

$$\tilde{g}(\mathcal{S}) = \frac{f(\mathcal{S})}{\sqrt{|\mathcal{A}| \cdot |\mathcal{B}|}}.$$

While there may be cases where this metric is successful, this metric is much less robust against high-degree nodes, as we next show. Note that degree distributions in real graphs are known to be highly skewed [Strogatz 2001]; in the extreme, Bhamidi et al. [2015] found that in certain large Twitter graphs, the largest degree nodes may be connected to a constant fraction of the number of nodes in the graph, in some cases exceeding half the total number of nodes. Hence, given a high degree node of degree cn, setting $\mathcal{S}$ as this node and its neighborhood gives a value of

$$\tilde{g}(\mathcal{S}) = \frac{cn}{\sqrt{1 \cdot cn}} = \sqrt{cn}, \tag{4}$$

proportional to $\sqrt{n}$. Note that if we instead consider a set $\mathcal{S}$ whose nodes all have average degree bounded by d, we have

$$\frac{f(\mathcal{S})}{\sqrt{|\mathcal{A}| \cdot |\mathcal{B}|}} \leq \frac{d \min(|\mathcal{A}|, |\mathcal{B}|)}{\sqrt{\min(|\mathcal{A}|, |\mathcal{B}|)^2}} = d. \tag{5}$$

Hence, consider a large, sparse graph. Due to the presence of skewed degree distributions [Strogatz 2001], we expect the vast majority of nodes to have relatively low average degree, which we model as being bounded by a constant d. As Bhamidi et al.

[2015] suggests, we may additionally expect a small number of very high degree nodes (with degree cn). As we have shown, the value of $\tilde{g}(\mathcal{S})$ on the maximum degree node and its neighborhood is proportional to $\sqrt{n}$, while $\tilde{g}(\mathcal{S})$ on typical sets $\mathcal{S}$ involving the majority of low average degree nodes will be close to bounded by d, which remains constant as n grows.

This suggests that in such cases, this metric is prone to attaining its maximum value when either $\mathcal{A}$ or $\mathcal{B}$ is a single node. In contrast, the normalization in the original metric $g(\mathcal{S}) = \frac{f(\mathcal{S})}{|\mathcal{S}|}$ is much less likely to pick up single nodes: e.g., if $|\mathcal{A}| = 1$, the value of $g(\mathcal{S})$ cannot exceed $\frac{|\mathcal{B}|}{1+|\mathcal{B}|} < 1$, an average degree of <1 that is likely to be exceeded by other subgraphs.

In conclusion, $g(\mathcal{S})$ can be interpreted as using a normalization term:

—*Scales linearly with the set size*: which, under simplified settings, can be seen as normalizing by the standard deviation.
—*Introduces a bias in favor of balanced sets $\mathcal{A}$ and $\mathcal{B}$*: in particular avoiding the "degenerate" case of catching sets consisting of a single high degree node (i.e., $|\mathcal{A}| = 1$ or $|\mathcal{B}| = 1$).

B. DETAILED ANALYSIS ON THE ACCURACY OF FRAUDAR

In this section, we provide a detailed analysis on the accuracy of FRAUDAR. Specifically, we give (1) necessary conditions and (2) sufficient conditions for FRAUDAR to detect fraudsters perfectly (i.e., with 100% precision and recall), as well as (3) necessary conditions and (4) sufficient conditions for FRAUDAR to detect at least one fraudulent node correctly.

We introduce the concept of coreness, which is used in the following analysis. Consider a subset $\mathcal{S}$ of $\mathcal{V}$. The weight of each node i in $\mathcal{S}$, denoted by $w_i(S)$, is defined as the sum of suspiciousness assigned to node i in the subgraph of G induced by $\mathcal{S}$, as in Definition 1. Based on this, we define the coreness of each node i, denoted by r_i, in Definition 2.

Definition 1 (*Weight in a Subgraph*). Let $\mathcal{S}$ be a subset of $\mathcal{V}$. For each node $i \in \mathcal{S}$, $w_i(\mathcal{S})$ indicates the sum of suspiciousness assigned to i in the subgraph induced by $\mathcal{S}$. That is,

$$w_i(\mathcal{S}) := a_i + \sum_{(j\in\mathcal{S})\wedge((i,j)\in\mathcal{E})} c_{ij} + \sum_{(j\in\mathcal{S})\wedge((j,i)\in\mathcal{E})} c_{ji},$$

where $a_i(\geq 0)$ indicates the weight of node i and $c_{ij}(>0)$ indicates that of edge (i, j) as in (2).

Definition 2 (*Coreness*). Each node $i \in \mathcal{V}$ has coreness r_i if and only if (1) there exists a subset $\mathcal{S} \subset \mathcal{V}$ such that $i \in \mathcal{S}$ and for every node $j \in \mathcal{S}$, $w_j(\mathcal{S}) \geq r_i$; and (2) there does not exist any subset $\mathcal{S} \subset \mathcal{V}$ such that $i \in \mathcal{S}$ and for every node $j \in \mathcal{S}$, $w_j(\mathcal{S}) > r_i$.

Having defined weight and coreness, we show the relation between them and FRAUDAR. The relation is used in the following analysis. Specifically, we prove that in each iteration of FRAUDAR, a node with minimum weight in the current subgraph is removed. We also prove that nodes are removed in a non-decreasing order of coreness.

In the rest of this section, we let π_t where $t \in \{1, 2, \ldots, n+m\}$ be the node removed in the tth iteration of Algorithm 1. We also let $\mathcal{X}_0 = \mathcal{V}$ and let $\mathcal{X}_t = \{\pi_h : 1 \leq t < h \leq n+m\}$ be the set of nodes removed in the tth or later iterations, as in Algorithm 1. In other words, each node π_t belongs to $\mathcal{X}_{t-1}$ but not to $\mathcal{X}_t$. We use $\pi_i^{-1} \in \{1, 2, \ldots, n+m\}$ to indicate the iteration in Algorithm 1 when node $i \in \mathcal{V}$ is removed.

LEMMA B.1. *In each iteration of Algorithm 1, a node with minimum weight in the current subgraph is removed. That is,*

$$w_{\pi_t}(\mathcal{X}_{t-1}) \leq w_j(\mathcal{X}_{t-1}),\ \forall j \in \mathcal{X}_{\pi_{t-1}}, \forall t \in \{1, 2, \ldots, n+m\}. \tag{6}$$

PROOF. This is reduced from the following condition in line 5 where $i^* = \pi_t$:

$$\frac{f(\mathcal{X}_{t-1}) - w_{i^*}(\mathcal{X}_{t-1})}{|\mathcal{X}_{t-1}| - 1} = g(\mathcal{X}_{t-1} \setminus \{i^*\}) \geq g(\mathcal{X}_{t-1} \setminus \{j\}) = \frac{f(\mathcal{X}_{t-1}) - w_j(\mathcal{X}_{t-1})}{|\mathcal{X}_{t-1}| - 1}, \forall j \in \mathcal{X}_{t-1}. \quad \square$$

LEMMA B.2. *The coreness of the node removed in each iteration of Algorithm 1 is equal to the maximum weight of nodes removed in the current or previous iterations. That is,*

$$r_{\pi_t} = \max\{w_{\pi_h}(\mathcal{X}_{h-1}) : 1 \leq h \leq t\},\ \forall t \in \{1, 2, \ldots, n+m\}. \tag{7}$$

PROOF. Let $w_t^* = \max\{w_{\pi_h}(\mathcal{X}_{h-1}) : 1 \leq h \leq t\}$. We prove $r_{\pi_t} = w_t^*$ by showing that $r_{\pi_t} \geq w_t^*$ and $r_{\pi_t} \leq w_t^*$. For each iteration t, let $h^* \leq t$ be the iteration where $w_{\pi_{h^*}}(\mathcal{X}_{h^*-1}) = w_t^*$. Then, in $\mathcal{X}_{h^*-1}$, which includes π_t, every node has weight at least $w_{\pi_{h^*}}(\mathcal{X}_{h^*-1})$ by Equation (6). Thus, by the definition of coreness, $r_{\pi_t} \geq w_{\pi_{h^*}}(\mathcal{X}_{h^*-1}) = w_t^*$. We also prove $r_{\pi_t} \leq w_t^*$ by showing that any subset of $\mathcal{V}$ where every node has weight strictly greater than w_t^* does not include π_t. Assume that such a subset $S \subset \mathcal{V}$ exists, and let $h' = \min\{h : \pi_h \in \mathcal{S}\}$ be the first iteration when a node in $\mathcal{S}$ is removed. From $\mathcal{S} \subset \mathcal{X}_{h'-1}$, we have $w_{\pi_{h'}}(\mathcal{X}_{h'-1}) \geq w_{\pi_{h'}}(\mathcal{S}) > w_t^*$. Since for every $t' \leq t$, $w_{\pi_{t'}}(\mathcal{X}_{t'-1}) \leq w_t^*$ by the definition of w_t^*, we have $t < h' = \min\{h : \pi_h \in \mathcal{S}\}$, and thus $\pi_t \notin \mathcal{S}$. Thus, by the definition of coreness, $r_{\pi_t} \leq w_t^*$. From $r_{\pi_t} \geq w_t^*$ and $r_{\pi_t} \leq w_t^*$, we have $r_{\pi_t} = w_t^*$. $\square$

LEMMA B.3. *In Algorithm 1, nodes are removed in a non-decreasing order of coreness, i.e.,*

$$\text{if } t < h,\ \text{then } r_{\pi_t} \leq r_{\pi_h},\ \forall t, h \in \{1, 2, \ldots, n+m\}.$$

PROOF. This lemma follows from Lemma B.2. $\square$

Assume $\mathcal{V}$ is divided into the set of fraudulent nodes $\mathcal{F}$ and the set of honest nodes $\bar{\mathcal{F}} = \mathcal{V} \setminus \mathcal{F}$. We say FRAUDAR detects fraudsters "perfectly" if it returns exactly $\mathcal{F}$. We give a necessary condition (Theorem B.4) and a sufficient condition (Theorem B.5) for this perfect detection. Let $\mathcal{X}_{t^*}$ be the set returned by FRAUDAR. In Theorem B.5, if (C1) and (C2) are met $\mathcal{X}_{t^*} \subset \mathcal{F}$ holds and fraudulent nodes are detected with 100% precision. If (C1) and (C3) are met, $\mathcal{X}_{t^*} \supset \mathcal{F}$ holds and 100% recall is achieved. If all conditions are met, $\mathcal{X}_{t^*} = \mathcal{F}$ holds and both 100% precision and recall are achieved.

THEOREM B.4 (NECESSARY CONDITION FOR PERFECT DETECTION). *Algorithm 1 returns $\mathcal{F}$ only if all fraudulent nodes have higher or equal coreness than honest nodes. That is,*

$$\forall u \in \mathcal{F}, \forall v \in \bar{\mathcal{F}}, r_u \geq r_v.$$

PROOF. Assume there exist $u \in \mathcal{F}$ and $v \in \bar{\mathcal{F}}$ such that $r_u < r_v$. By Lemma B.3, we have $\pi_u^{-1} < \pi_v^{-1}$. Then, for every $t < \pi_v^{-1}$, we have $v \in \mathcal{X}_t$, and for every $t \geq \pi_v^{-1}$, we have $u \notin \mathcal{X}_t$. Thus, $\mathcal{F}$ is not included in $\{\mathcal{X}_t : 0 \leq t \leq n+m\}$, among which Algorithm 1 returns one. Hence, such $u \in \mathcal{F}$ and $v \in \bar{\mathcal{F}}$ should not exist for Algorithm 1 to return $\mathcal{F}$. $\square$

THEOREM B.5 (SUFFICIENT CONDITION FOR PERFECT DETECTION). *Let $r^*_{\bar{\mathcal{F}}} = \{r_v : v \in \bar{\mathcal{F}}\}$ be the maximum coreness among honest nodes. Algorithm 1 returns $\mathcal{F}$ if all the following conditions are met: (C1) all fraudulent nodes have strictly higher coreness than honest nodes, i.e., $\forall u \in \mathcal{F}$, $r_u > r^*_{\bar{\mathcal{F}}}$, (C2) fraudulent nodes have density strictly higher than $r^*_{\bar{\mathcal{F}}}$, i.e., $g(\mathcal{F}) > r^*_{\bar{\mathcal{F}}}$, and (C3) all proper subsets of $\mathcal{F}$ have strictly lower density than $\mathcal{F}$, i.e., $\forall \mathcal{S} \subsetneq \mathcal{F}, g(\mathcal{S}) < g(\mathcal{F})$.*

PROOF. By (C1) and Lemma B.3, there exists $t \in \{1, 2, \ldots, n+m\}$ where $\mathcal{X}_t = \mathcal{F}$. By (C3), for every $h > t$, $g(\mathcal{X}_h) < g(\mathcal{X}_t)$ since $\mathcal{X}_h$ is a proper subset of $\mathcal{X}_t = \mathcal{F}$. Moreover, for every $h < t$, $g(\mathcal{X}_h) < g(\mathcal{X}_t) = g(\mathcal{F})$ since

$$\begin{aligned} g(\mathcal{X}_h) &= \frac{f(|\mathcal{F}|) + \sum_{s=h+1}^{t} w_{\pi_s}(\mathcal{X}_{s-1})}{|F| + t - h} \leq \frac{f(|\mathcal{F}|) + \sum_{s=h+1}^{t} r_{\pi_s}}{|F| + t - h} \\ &\leq \frac{f(|\mathcal{F}|) + (t-h) r^*_{\bar{\mathcal{F}}}}{|F| + t - h} < \frac{f(|\mathcal{F}|) + (t-h) g(\mathcal{F})}{|F| + t - h} \qquad \text{(By (C2))} \\ &= \frac{f(|\mathcal{F}|) + (t-h) f(\mathcal{F})/|F|}{|F| + t - h} = \frac{f(\mathcal{F})}{|F|} = g(\mathcal{F}). \end{aligned}$$

Hence, $\mathcal{F} = \mathcal{X}_t = \arg\max_{\mathcal{X}_h \in \{\mathcal{X}_0, \ldots, \mathcal{X}_{m+n}\}} g(\mathcal{X}_h)$, which is returned by Algorithm 1. □

On the other hand, we say FRAUDAR detects fraudsters "marginally" if $\mathcal{X}_{t^*}$, the set returned by Algorithm 1, includes at least one fraudulent node (i.e., $\mathcal{X}_{t^*} \cap \mathcal{F} \neq \emptyset$). We give a necessary condition (Theorem B.6) and a sufficient condition (Theorem B.7) for this marginal detection.

THEOREM B.6 (NECESSARY CONDITION FOR MARGINAL DETECTION). *Let $r^*_{\mathcal{F}} = \{r_v : v \in \mathcal{F}\}$ be the maximum coreness among fraudulent nodes. Likewise, we define $r^*_{\bar{\mathcal{F}}} = \{r_v : v \in \bar{\mathcal{F}}\}$ for honest nodes. If we let $\mathcal{X}_{t^*}$ be the set returned by Algorithm 1, $\mathcal{X}_{t^*} \cap \mathcal{F} \neq \emptyset$ only if $r^*_{\mathcal{F}} \geq r^*_{\bar{\mathcal{F}}}/2$. That is, $r^*_{\mathcal{F}} \geq r^*_{\bar{\mathcal{F}}}/2$ is a necessary condition for* FRAUDAR *to return at least one fraudulent node.*

PROOF. By Lemma B.1 and Lemma B.2, there exists $t' \in \{0, 1, \ldots, n+m\}$ such that every node in $\mathcal{X}_{t'}$ has weight at least $r^*_{\bar{\mathcal{F}}}$ (i.e., $\forall i \in \mathcal{X}_{t'}$, $w_i(\mathcal{X}_{t'}) \geq r^*_{\bar{\mathcal{F}}}$). Then,

$$g(\mathcal{X}_{t^*}) \geq g(\mathcal{X}_{t'}) = \frac{f(\mathcal{X}_{t'})}{|\mathcal{X}_{t'}|} \geq \frac{\sum_{i \in \mathcal{X}_{t'}} w_i(\mathcal{X}_{t'})/2}{|\mathcal{X}_{t'}|} \geq \frac{\sum_{i \in \mathcal{X}_{t'}} r^*_{\bar{\mathcal{F}}}/2}{|\mathcal{X}_{t'}|} = r^*_{\bar{\mathcal{F}}}/2.$$

From this, it follows that every nodes in $\mathcal{X}_{t^*}$ has weight at least $r^*_{\bar{\mathcal{F}}}/2$. That is,

$$w_i(\mathcal{X}_{t^*}) \geq r^*_{\bar{\mathcal{F}}}/2, \ \forall i \in \mathcal{X}_{t^*} \tag{8}$$

This is since, otherwise,

$$g(\mathcal{X}_{t^*+1}) = \frac{f(\mathcal{X}_{t^*}) - w_{\pi_{t^*+1}}(\mathcal{X}_{t^*})}{|\mathcal{X}_{t^*}| - 1} > \frac{f(\mathcal{X}_{t^*}) - r^*_{\bar{\mathcal{F}}}/2}{|\mathcal{X}_{t^*}| - 1} \geq \frac{f(\mathcal{X}_{t^*}) - g(\mathcal{X}_{t^*})}{|\mathcal{X}_{t^*}| - 1} \geq g(\mathcal{X}_{t^*}),$$

which is a contradiction because $\mathcal{X}_{t^*} = \arg\max_{\mathcal{X}_t \in \{\mathcal{X}_0, \ldots, \mathcal{X}_{m+n}\}} g(\mathcal{X}_t)$.

Since $r^*_{\mathcal{F}}$ is the maximum coreness among fraudulent nodes, every $\mathcal{X}_t \in \{\mathcal{X}_0, \ldots, \mathcal{X}_{m+n}\}$ including at least one fraudulent node (i.e., $\mathcal{X}_t \cap \mathcal{F} \neq \emptyset$) has a node with weight at most $r^*_{\mathcal{F}}$. Thus, $\mathcal{X}_{t^*} \cap \mathcal{F} \neq \emptyset$ can be true only when $r^*_{\mathcal{F}} \geq \min_{i \in \mathcal{X}_{t^*}} w_i(\mathcal{X}_{t^*}) \geq r^*_{\bar{\mathcal{F}}}/2$ by Equation (8). Hence, $r^*_{\mathcal{F}} \geq r^*_{\bar{\mathcal{F}}}/2$ is a necessary condition for $\mathcal{X}_{t^*} \cap \mathcal{F} \neq \emptyset$. □

THEOREM B.7 (SUFFICIENT CONDITION FOR MARGINAL DETECTION). *Let $g^*_{\mathcal{F}} = \max_{\mathcal{S} \subset \mathcal{F}} g(\mathcal{S})$ be the maximum density achievable by subsets purely consisting of fraudulent nodes. Likewise, we define $g^*_{\bar{\mathcal{F}}} = \max_{\mathcal{S} \subset \bar{\mathcal{F}}} g(\mathcal{S})$ for honest nodes. If we let $\mathcal{X}_{t^*}$ be the set returned by Algorithm 1, $\mathcal{X}_{t^*} \cap \mathcal{F} \neq \emptyset$ if $g^*_{\mathcal{F}} > 2g^*_{\bar{\mathcal{F}}}$. That is, $g^*_{\mathcal{F}} > 2g^*_{\bar{\mathcal{F}}}$ is a sufficient condition for* FRAUDAR *to return at least one fraudulent node.*

PROOF. Let $g^* = \max_{\mathcal{S} \subset \mathcal{V}} g(\mathcal{S})$ be the maximum density. If $g^*_{\mathcal{F}} > 2g^*_{\bar{\mathcal{F}}}$, we have $g^* \geq g^*_{\mathcal{F}} > 2g^*_{\bar{\mathcal{F}}}$. By Theorem 4.3, $g(\mathcal{X}_{t^*}) \geq g^*/2 > g^*_{\bar{\mathcal{F}}}$. Hence, $\mathcal{X}_{t^*} \not\subseteq \bar{\mathcal{F}}$ and thus $\mathcal{X}_{t^*} \cap \mathcal{F} \neq \emptyset$. □

REFERENCES

Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion fraud detection in online reviews by network effects. In *Proceedings of the 7th International AAAI Conference on Weblogs and Social Media*.

Alex Beutel, Kenton Murray, Christos Faloutsos, and Alexander J. Smola. 2014. Cobafi: Collaborative Bayesian filtering. In *Proceedings of the 23rd International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 97–108.

Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. Copycatch: Stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee. 119–130.

Shankar Bhamidi, J. Michael Steele, Tauhid Zaman, and others. 2015. Twitter event networks and the superstar model. *The Annals of Applied Probability* 25, 5 (2015), 2462–2502.

Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. 2012. Aiding the detection of fake accounts in large scale social online services. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*.

Moses Charikar. 2000. Greedy approximation algorithms for finding dense components in a graph. In *Approximation Algorithms for Combinatorial Optimization*. Springer, 84–95.

Corinna Cortes, Daryl Pregibon, and Chris Volinsky. 2001. *Communities of Interest*. Springer.

Saptarshi Ghosh, Bimal Viswanath, Farshad Kooti, Naveen Kumar Sharma, Gautam Korlam, Fabricio Benevenuto, Niloy Ganguly, and Krishna Phani Gummadi. 2012. Understanding and combating link farming in the twitter social network. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 61–70.

Christos Giatsidis, Dimitrios M. Thilikos, and Michalis Vazirgiannis. 2011. Evaluating cooperation in communities with the k-core structure. In *Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 87–93.

Zhongshu Gu, Kexin Pei, Qifan Wang, Luo Si, Xiangyu Zhang, and Dongyan Xu. 2015. LEAPS: Detecting camouflaged attacks with statistical learning guided by program analysis. In *Proceedings of 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 57–68.

Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. 2004. Combating web spam with trustrank. In *Proceedings of the 30th International Conference on Very Large Data Bases*. 576–587.

Meng Jiang, Alex Beutel, Peng Cui, Bryan Hooi, Shiqiang Yang, and Christos Faloutsos. 2015. A general suspiciousness metric for dense blocks in multimodal data. In *Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM)*. IEEE, 781–786.

Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014b. CatchSync: Catching synchronized behavior in large directed graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 941–950.

Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014a. Inferring strange behavior from connectivity pattern in social networks. In *Advances in Knowledge Discovery and Data Mining*. Springer, 126–138.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining 2008*. ACM, 219–230.

Chris Kanich, Christian Kreibich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. 2008. Spamalytics: An empirical analysis of spam marketing conversion. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*. ACM, 3–14.

Chris Kanich, Nicholas Weaver, Damon McCoy, Tristan Halvorson, Christian Kreibich, Kirill Levchenko, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. 2011. Show me the money: Characterizing spam-advertised revenue. In *Proceedings of the 20th USENIX Security Symposium*. 15–15.

G. Karypis and V. Kumar. 1995. METIS: Unstructured graph partitioning and sparse matrix ordering system, Version 2. The University of Minnesota.

J. M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46, 5 (1999), 604–632.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 591–600.

Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Signed networks in social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1361–1370.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*. ACM, 165–172.

Bhaskar Mehta and Thomas Hofmann. 2008. A survey of attack-resistant collaborative filtering algorithms. *IEEE Technical Committee on Data Engineering* 31, 2 (2008), 14–22.

Bhaskar Mehta, Thomas Hofmann, and Wolfgang Nejdl. 2007. Robust collaborative filtering. In *Proceedings of the 2007 ACM Conference on Recommender Systems*. ACM, 49–56.

Bhaskar Mehta and Wolfgang Nejdl. 2008. Attack resistant collaborative filtering. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 75–82.

Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology (TOIT)* 7, 4 (2007), 23.

Arash Molavi Kakhki, Chloe Kliman-Silver, and Alan Mislove. 2013. Iolaus: Securing online content rating systems. In *Proceedings of the 22nd International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 919–930.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming* 14, 1 (1978), 265–294.

Michael O'Mahony, Neil Hurley, Nicholas Kushmerick, and Guénolé Silvestre. 2004. Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technology (TOIT)* 4, 4 (2004), 344–377.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Vol. 1, Association for Computational Linguistics, 309–319.

Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 2007. Netprobe: A fast and scalable system for fraud detection in online auction networks. In *Proceedings of the 16th International Conference on World Wide Web*. ACM, 201–210.

Bryan Perozzi, Leman Akoglu, Patricia Iglesias Sánchez, and Emmanuel Müller. 2014. Focused clustering and outlier detection in large attributed graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1346–1355.

B. A. Prakash, M. Seshadri, A. Sridharan, S. Machiraju, and C. Faloutsos. 2010. Eigenspokes: Surprising patterns and community structure in large graphs. *Pacific Asia Knowledge Discovery and Data Mining, 2010a.* Vol. 84.

Anand Rajaraman, Jeffrey D. Ullman, Jeffrey David Ullman, and Jeffrey David Ullman. 2012. *Mining of Massive Datasets*. Vol. 1, Cambridge University Press Cambridge.

Neil Shah, Alex Beutel, Brian Gallagher, and Christos Faloutsos. 2014. Spotting suspicious link behavior with fBox: An adversarial perspective. In *Proceedings of the 2014 IEEE International Conference on Data Mining (ICDM'14)*. IEEE, 959–964.

Gianluca Stringhini, Manuel Egele, Christopher Kruegel, and Giovanni Vigna. 2012. Poultry markets: On the underground economy of twitter followers. In *Proceedings of the 2012 ACM Workshop on Workshop on Online Social Networks*. ACM, 1–6.

Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. 2010. Detecting spammers on social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*. ACM, 1–9.

Steven H. Strogatz. 2001. Exploring complex networks. *Nature* 410, 6825 (2001), 268–276.

Dinh Nguyen Tran, Bonan Min, Jinyang Li, and Lakshminarayanan Subramanian. 2009. Sybil-resilient online content voting. In *Proceedings of the 6th USENIX symposium on Networked Systems Design and Implementation*, Vol. 9, 15–28.

Charalampos Tsourakakis. 2015. The K-clique densest subgraph problem. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1122–1132.

Sankar Virdhagriswaran and Gordon Dakin. 2006. Camouflaged fraud detection in domains with complex relationships. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 941–947.

Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 618–626.

Steve Webb, James Caverlee, and Calton Pu. 2008. Social honeypots: Making friends with a spammer near you. In *Conference Proceedings of on Email and Anti-Spam*.

Baoning Wu, Vinay Goel, and Brian D. Davison. 2006. Propagating trust and distrust to demote web spam. In *Proceedings of the Workshop on Models of Trust for the Web*. Vol. 190.

Haifeng Yu, Phillip B. Gibbons, Michael Kaminsky, and Feng Xiao. 2008. Sybillimit: A near-optimal social network defense against sybil attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 3–17.

Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. 2006. Sybilguard: Defending against sybil attacks via social networks. *ACM SIGCOMM Computer Communication Review* 36, 4 (2006), 267–278.

Received November 2016; revised Januray 2017; accepted February 2017