Hindawi
Mobile Information Systems
Volume 2017, Article ID 6412521, 12 pages
https://doi.org/10.1155/2017/6412521

Research Article

iBGP: A Bipartite Graph Propagation Approach for Mobile Advertising Fraud Detection

Jinlong Hu, Junjie Liang, and Shoubin Dong

School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Correspondence should be addressed to Jinlong Hu; jlhu@scut.edu.cn

Received 23 February 2017; Accepted 13 March 2017; Published 3 April 2017

Academic Editor: Elio Masciari

Online mobile advertising plays a vital financial role in supporting free mobile apps, but detecting malicious apps publishers who generate fraudulent actions on the advertisements hosted on their apps is difficult, since fraudulent traffic often mimics behaviors of legitimate users and evolves rapidly. In this paper, we propose a novel bipartite graph-based propagation approach, iBGP, for mobile apps advertising fraud detection in large advertising system. We exploit the characteristics of mobile advertising user's behavior and identify two persistent patterns: *power law distribution* and *pertinence* and propose an automatic initial score learning algorithm to formulate both concepts to learn the initial scores of non-seed nodes. We propose a weighted graph propagation algorithm to propagate the scores of all nodes in the user-app bipartite graphs until convergence. To extend our approach for large-scale settings, we decompose the objective function of the initial score learning model into separate one-dimensional problems and parallelize the whole approach on an Apache Spark cluster. iBGP was applied on a large synthetic dataset and a large real-world mobile advertising dataset; experiment results demonstrate that iBGP significantly outperforms other popular graph-based propagation methods.

1. Introduction

Online mobile advertising plays a vital financial role in supporting free mobile apps. The mobile advertising service platform is the main coordinator, acting as a broker between advertisers and content publishers (typically an app owner). Advertisers pay advertising service platforms for each customer action (e.g., clicking an ad, filling a form, and downloading and installing an app), and advertising service platforms pay publishers a fraction of the revenue for each customer action on their apps. However, this pay-per-click or pay-per-action model may incentivize malicious publishers to generate fraudulent actions on the advertisements to get more financial returns. This issue, similar to *click fraud*, has been a serious threat for online advertising market over the years [1]. Thus, it is important to develop a reliable fraud detection system that can monitor a publisher's behavior and efficiently identify whether a publisher is likely to be fraudulent.

Fraud detection in online mobile advertising (i.e., detecting fraudulent app publishers who unfairly bolster their volume of actions) is a challenging task, not only because fraudulent traffic often mimics that of legitimate customers but also due to the rapid evolving fraud techniques. Traditional fraud detection methods, for example, rule-based systems [2], could usually be effective in filtering out these fraudulent behaviors, when the specific characteristics of fraudsters' behavior patterns (e.g., repetitive clicks or hit bursts) were well-studied and the detection rules were appropriately defined. Unfortunately, fraudsters often adjust their fraudulent behaviors accordingly to escape the predefined rules, so traditional fraud detection systems are usually difficult to adapt to novel anomalies as well as the changing and growing data in face of adversaries [3].

In recent years, graph-based propagation methods for fraud detection are tried in several areas [4–9], for their relational nature of the problem domain, adversarial robustness, and other graph-based advantages [3]. These methods, working in an unsupervised fashion, perform propagation starting from known trust/distrust scores of nodes (seeds) and update all nodes (both seeds and nonseeds) iteratively until some convergence criterion is reached, which could

generally achieve higher accuracy compared to the traditional methods in detecting fraudulent behaviors. Such methods often explicitly or implicitly assign a certain value as initial scores for non-seed nodes prior to the propagation phase. However, the initial scores of nodes could usually affect the convergent results of propagation in graph [10], and, therefore, getting more seeds with labels from the large system is crucial to enhance accuracy. Practically, to find the fraudsters or obtain labels (scores) of nodes is labor-intensive in large online mobile advertising system, and, in most scenarios, only a rare number of seeds with labels are achievable, which indicates the non-seed nodes without labels would far outnumber those of the seeds with labels.

To address the above challenges, we propose a novel graph-based propagation approach for online mobile advertising fraud detection, which introduces an automatic initial score learning algorithm that utilizes the side information in a large user-app bipartite graph propagation method. The proposed approach shows both effectiveness and efficiency in fraudulent apps detection over a real-world online mobile advertising dataset and a synthetic dataset. In this paper, we first exploit the characteristics of mobile advertising users behavior and identify two persistent patterns: (a) *power law distribution*: the fraud scores of the majority of users follow the same patterns while very few of them fall in the tail, which properly fits to the power law distribution, and (b) *pertinence*: the distributions of users' targeting behaviors in a given period are sharply skewed. Then we proposed a novel approach called iBGP (bipartite graph propagation with initial score learning), which consists of three stages: (a) graph constructing stage: a user-app bipartite weighted graph is constructed based on user behavior logs; (b) initial score learning stage: the initial scores of seeds and nonseeds are learned separately through empirical analysis on the side information; (c) propagation stage: a weighted HITS algorithm is used to propagate the scores of all nodes in the large user-app bipartite graph.

Our contributions could be summarized as follows:

(i) We propose iBGP, a new graph-based propagation approach with initial score learning for fraud detection in mobile advertising. To the best of our knowledge, this is the first work to integrate the initial scores learning algorithm for non-seed nodes with side information into a graph-based propagation method, which would significantly improve the accuracy of propagation on the large system where precise labels are rare.

(ii) We identify two behavior patterns of the fraudsters *(power law distribution* and *pertinence)* and mathematically formulate both patterns into an integrated model, which is in return used to determine the initial scores of non-seed nodes.

(iii) We parallelize the initial scores learning algorithm by decomposing the objective function into separate one-dimensional problems and further implement the approach on an Apache Spark cluster to extend our method to large-scale bipartite graphs.

We evaluate our approach on a large synthetic dataset and a large real-world dataset from one of the mobile advertising platforms in China. Results show that we effectively detect fraudulent apps with high accuracy, which is superior to the popular traditional graph propagation methods and their adaptations. The rest of the paper is organized as follows. Section 2 discusses related work. We formulate our problem and present our model in Section 3, and Section 4 reports on experiments. We conclude the paper in Section 5.

2. Related Work

Our work is related to existing studies on graph-based fraud detection and click fraud detection. As stated in [11], the challenges of the click fraud detection problem for online advertising are summarized as rapidity of model updates needed to combat attackers and programmability of attacks and accuracy requirements. Metwally et al. [2] introduce streaming-rules with tight guarantees on errors, in order to detect fraud caused by malwares, autoclickers, and so forth. Unfortunately, rule-based methods are labor-intensive and could soon be invalid due to the rapid evolvement of fraudsters, and there is no universal method that can detect all kinds of frauds at the same time [12].

In recent years, graph-based anomaly detection methods are widely studied in many research areas due to their advantages on interdependent nature of the data, powerful representation, relational nature of problem domains, and robust machinery [3]. In particular, several graph-based propagation methods are tried for fraud detection, for example, biased PageRank, that is, TrustRank, DistrustRank, and their integration [5, 6, 9]. In these models, a set of highly trustful/distrustful sites (seeds) are chosen and initial scores associated with their labels are assigned by either human experts or empirical studies; then the biased PageRank methodology is adopted to propagate these scores to the entire graph iteratively until convergence. As for bipartite graphs, methods based on the popular Kleinberg's HITS algorithms [13] are applied. Li et al. [8] adapt the HITS model to detect session-level cheating, where the fraud scores of user nodes are fixed to one and only the scores of other nodes are updated during the propagation. Dai et al. [4] explore both positive and negative dependencies and encode the anomalous scores to the edges between source and target nodes with the intuition to propagate anomaly through both parts. Also related are the works of Belief Propagation (BP) [14–16], where multiple states of nodes are predefined in a Markov Random Field, and the likelihood of each state within the nodes can be computed using the propagation matrix. In the first propagation pass, the non-seed nodes of these methods (usually with unknown initial scores) are either explicitly or implicitly assigned a certain initial value (i.e., 0.1 or 0). However, Agosti and Pretto [10] prove that convergent results of both HITS and its adaptations are associated with the initial scores of nodes; therefore, initial scores without careful examination might significantly deteriorate the advanced model with well-designed propagation methodology.

In this paper, we propose a novel approach to the propagation algorithm with the initial scores learning method for mobile advertising fraud detection in bipartite graphs, based on the user behavior patterns and their background distribution.

3. Mobile Ad Fraud Detection

In this section, we formally present the problem definition of mobile ad fraud detection and then propose an effective solution.

3.1. Problem Definition. Our goal is to find fraudulent apps on a user-app undirected bipartite graph, and the problem could be defined as follows.

Given. An undirected bipartite graph $\mathcal{G} = \langle \mathcal{U} \cup \mathcal{A}, E, W \rangle$, where $\mathcal{U}$ is the source or user nodes and $\mathcal{A}$ is the target or app nodes, $E \subset \mathcal{U} \times \mathcal{A}$ is a set of undirected edges between the users and the apps, and $W = \{w_{ua}\}$ is a set of edge weights. (See Figure 1(b) for an example.)

Find. A set of suspicious app nodes $\mathcal{A}_f$ whose fraud scores are relatively high. The definitions of symbols throughout this paper are listed in Symbols and Definitions.

3.2. Proposed Approach. In this section, we introduce iBGP to address the aforementioned problem. First, we provide the constructing stage of user-app bipartite graph. Second, we present the propagating stage. We partition the users by seeds and nonseeds as in [5, 6, 8, 9, 17, 18], while, in this paper, the seeds are determined by an outlier detection method and scores of nonseeds are learned by a user behavior model. Propagation is performed after initial scores of both seeds and nonseeds are assigned.

3.2.1. Constructing User-App Bipartite Graph. We collect user behavior logs from a mobile advertising platform, which maintains a history of user actions that happened within a time period, including viewing, clicking, download start, download completion, installation start, installation completion. The following attributes are studied: *user ID*: an id to identify a unique user; *app ID*: an id to identify a unique app; *geographical attributes*: a series of user geographical attributes are used to detect anomalies, including encrypted IP and city; *action time*: it is the timestamp when the action happened; *mobile attributes*: certain characteristics of user device are also studied, for example, device ID, device system models, and screen size. A seven-day (2015.6.1–2015.6.7) mobile advertising user behavior log is studied. Some examples of our raw data are shown in Figure 1(a).

Let $\mathcal{U}$ be the source (users) and $\mathcal{A}$ be the target (apps); we form an edge e_{ua} from user u to app a if there exists an action from u to a, such that $E \subset U \times A$ is the set of edges from the source to the target. The set of edge weights $W = \{w_{ua}\}$ is defined proportional to the behavioral centrality of u to a, such that an undirected graph $\mathcal{G} = \langle \mathcal{U} \cup \mathcal{A}, E, W \rangle$ is built as stated in Figure 1(b).

3.2.2. Propagating User Scores in Bipartite Graph. As stated in the prior section, initial scores of users should be determined before propagation. We first discuss the determination of initial user scores, and then we present the propagation process.

Detecting Outlier Users as Seeds. We start by performing a domain knowledge based feature selection. Empirically, we aim to find the users that are too far away from the majority. Hence, it is straightforward to define the suspiciousness of user u by how many predictors of u are λ times standard deviation away from the mean:

$$\mathcal{S}_u = \sum_{i=1}^{P} I\left\{u^{(i)} \geq \mu_i + \lambda\sigma_i\right\}, \tag{1}$$

where P is the number of predictors. We assign $\lambda = 3.0$ and $\mathcal{U}_f = \{u \mid \mathcal{S}_u \geq P/2\}$ such that a relatively small proportion of users are eventually tagged as fraudsters. In our dataset, approximately 6% of users are flagged each day.

Computing Initial Scores of Non-Seed Users. We develop a probabilistic model that combines power law and user pertinence. We present the iBGP model that is based on the following intuitions:

(i) *Power Law Distribution*. Scores of non-seed users are subject to a power law distribution.

(ii) *User Pertinence*. True fraudsters are extremely targeted, and initial score of user u can be estimated by u's targeting behaviors. The term "targeting" or "u targets at a" means u performed additional actions at a other than just viewing, for example, clicking, download start, or installation start.

We now describe these two components of the model in further detail.

Modeling Power Law Distribution. We group our logs by one-day period and perform data statistics on the user part. Similar scenarios are found regardless of days, and here we list the typical statistics of attributes on 1st June in Figure 2. Obviously, the majority of users follow the same patterns while very few of them fall in the tail, which can be largely described by power law distributions.

In order to model the power law distribution of user scores, we aim to capture two intuitions: (a) score of the user is subject to a power law distribution and (b) the majority of users are normal.

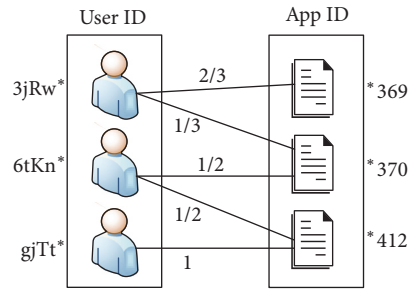
To achieve these goals, we assume the score of each node is drawn from a continuous probability density $P(x)$ such that

$$P(x) = Cx^{-\beta}, \tag{2}$$

where β ($\beta > 0$) is a constant parameter of the distribution known as exponent or scaling parameter. C ($C > 0$) is

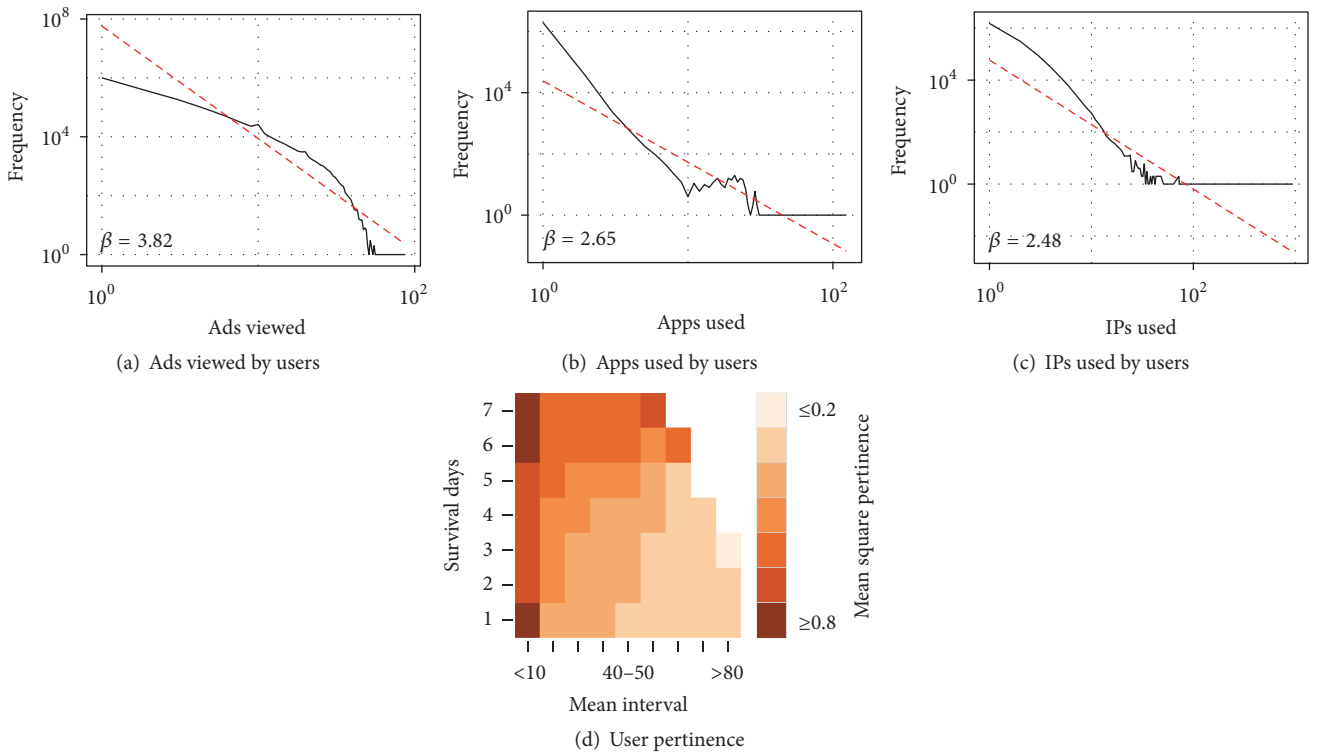
User ID	App ID	Action	⋯
3jRw*	*369	Viewing	⋯
3jRw*	*369	Clicking	⋯
3jRw*	*370	Viewing	⋯
6tKn*	*370	Download start	⋯
6tKn*	*412	Download completion	⋯
gjTt*	*412	Installation start	⋯
gjTt*	*412	Installation completion	⋯

(a) Raw data



(b) Bipartite graph

Figure 1: Examples of the raw data and the constructed user-app bipartite graph.



(a) Ads viewed by users (b) Apps used by users (c) IPs used by users

(d) User pertinence

Figure 2: Characteristics of mobile app users within a one-day period: the behavior patterns of mobile users can be generally described by power law distributions as shown in (a), (b), and (c). (d) Higher mean square pertinence tend to correlate with lower mean interval for non-seed users.

a normalization constant. Clearly this density diverges as $x \to 0$, and we denote a lower bound of x by δ ($\delta > 0$). Without loss of generality, we set the upper bound of x by 1. So we have $x \in [\delta, 1]$ with δ indicating absolute normality and 1 indicating absolute fraud, such that, $\forall u \in \mathcal{U}_f$, $x_u = 1$.

Let $F(x)$ be the distribution function of $P(x)$. We expand $F(x)$ by Taylor series at $x = 1$ and combine the constraints $F(\delta) = 0$ and $F(1) = 1$; we arrive at

$$P(x;\delta) = \left[\sum_{n=1}^{\infty} (-1)^n \frac{(\beta+n-2)!}{(\beta-1)!\cdot n!} (\delta-1)^n\right]^{-1} x^{-\beta}. \quad (3)$$

We assume that users are mutually independent. Hence, we can derive the log-likelihood $l_V(X) = \log P(X;\delta)$ as

$$l_V(X;\delta,\beta) = -\sum_{u\in\mathcal{U}_n} \left[\log\left(\sum_{n=1}^{\infty} (-1)^n \frac{(\beta+n-2)!}{(\beta-1)!\cdot n!} (\delta-1)^n\right) + \beta \log x_u\right]. \quad (4)$$

Clearly, (4) satisfies our two aforementioned intuitions.

Modeling User Pertinence with Power Law. We propose the novel concept "user pertinence" to investigate the characteristics of users' behavior patterns. Pertinence showcases the behavioral centrality of users to elucidate how evident the user u's targets are in a given period of time, such that, similar to the definition of w_{ua}, user pertinence between u and a can be formulated by the proportion of actions that u targets at a:

$$t_{ua} = \frac{|T_{ua}|}{\sum_{k \in \mathcal{O}(u)} |T_{uk}|}. \tag{5}$$

Note that $\sum_{a \in \mathcal{O}(u)} t_{ua} = 1$ holds for all users.

Commonly, fraudsters are motivated by monetary rewards and targeting behaviors require more in-depth actions compared to browsing. To explore the characteristic of user pertinence, we investigate the following indicators of seed nodes and non-seed nodes separately: (a) *mean interval*: the average time intervals between each of the first browsing and first targeting behavior of a user on seven days, (b) *survival days*: the number of days that the user exists in our logs, and (c) *mean square pertinence*: the average square pertinence of user over seven days. For seed nodes, the characteristic of user pertinence is clear: 78% of them survive only one day, among which 76% have mean interval below 10 seconds and mean square pertinence over 0.76. For non-seed nodes, we observe similar phenomenon as shown in Figure 2(d), where users with mean interval lower than 10 seconds showcase high mean square pertinence. Inspired by the characteristic of seed users, those nonseeds who share the same patterns with seeds are deemed highly suspicious. Moreover, mean square pertinence is more stable since mean interval correlates strongly to the fluctuation of network quality. As a result, we adopt user pertinence to predict the fraud scores of nonseeds.

It is straightforward to infer that a higher fraud score tends to associate with greater user pertinence and vice versa. We model this intuition with separate logistic models. For each $u \in \mathcal{U}_n$, we define the user score likelihood by

$$Q_u = \sigma(h_u), \tag{6}$$

where $\sigma(\cdot)$ is a sigmoid function and h_u represents the relevance between x_u and $\{t_{ua} \mid a \in \mathcal{O}(u)\}$. The form of h_u should catch the following requirements: (a) properly depicting the relationship between x_u and $\{t_{ua} \mid a \in \mathcal{O}(u)\}$ and (b) being differentiable and simple.

We adopt a linear function as

$$h_u = -6 \sum_{a \in \mathcal{O}(u)} t_{ua} |x_u - t_{ua}| + 3. \tag{7}$$

The definition of h_u satisfies our two requirements. The positive correlation between x_u and $\{t_{ua} \mid a \in \mathcal{O}(u)\}$ is defined by $\sum_{a \in \mathcal{O}(u)} t_{ua} |x_u - t_{ua}|$, which is differentiable for all $\{x_u \mid x_u \neq t_{ua}\}$ and simple (Requirement (b)). The coefficient t_{ua} in $t_{ua}|x_u - t_{ua}|$ denotes that greater user pertinence outweighs the weaker one (Requirement (a)). By assigning the gradient as -6 and interception as 3, Q_u is scaled close to $(0, 1)$, and symmetry approximately holds under a relatively small δ. Similarly, we derive the log-likelihood $l_G(X) = \log Q_u(X)$ as follows:

$$l_G(X; \delta) = -\sum_{u \in \mathcal{U}_n} \log\left(1 + \exp\left(-h_u\right)\right). \tag{8}$$

Finally, we aim to infer the optimal $\widehat{X}$ by maximizing the likelihood on both l_V and l_G. Therefore, the final problem could be organized as

$$\widehat{X} = \arg\max_X \mathcal{L}(X; \delta, \beta) = \arg\max_X \alpha l_V + (1 - \alpha) l_G, \tag{9}$$

where α is a regularization hyperparameter, defining the significance of power law distribution.

Computing Initial User Scores. Note that $\mathcal{L}$ is continuous on interval $[\delta, 1]$. Traditionally, $\mathcal{L}$ could be solved approximately by one of the gradient descent methods. However, these solving methods are computationally intensive when the dimension of $\mathcal{U}_n$ is ultrahigh. Since users are mutually independent in our model's assumption, we can further decompose (9) into separate one-dimensional problem on each user as

$$\begin{aligned} \widehat{x}_u &= \arg\min_{x_u} \mathcal{L}_u(x_u; \delta, \beta) \\ &= \arg\min_{x_u} \left(\alpha\beta \log x_u + (1 - \alpha) \log\left(1 + \exp\left(-h_u\right)\right)\right). \end{aligned} \tag{10}$$

Now $\mathcal{L}$ could be solved efficiently by parallel operators on subproblems in (10). We gain the approximate optimal solutions of $\mathcal{L}_u$ for all $u \in \mathcal{U}_n$ in a more effective way using Golden Section Method (GSM) [19], which is notably efficient in one-dimensional searching. The convergence of this method is guaranteed under the continuity of (10). A description of our method is shown in Algorithm 1.

Propagating User Scores. The basic assumption of propagating algorithm is that if a number of users of a certain app are fraudsters, the app itself is likely to be a cheating one as well. In accordance with the weighted HITS algorithm [13], we complete the propagation process as follows:

$$x_a = \frac{\sum_{u \in \mathcal{F}(a)} w_{ua} x_u}{\sum_{u \in \mathcal{F}(a)} w_{ua}}, \tag{11}$$

$$x_u = \sum_{a \in \mathcal{O}(u)} w_{ua} x_a. \tag{12}$$

A complete iteration of propagation on bipartite graph consists of two steps. Scores on user nodes are first propagated to the app nodes as in (11) and then propagated back to update the scores of user nodes as in (12). The iteration works successively until either the maximum iteration limit is reached or scores on user nodes converge.

Finally, we compute the ranking of app scores and flag the top-k results as fraud apps. Combining Algorithm 1, (11), and (12), the overall description of iBGP is listed in Algorithm 2.

Implementation of iBGP on Apache Spark Cluster. To extend iBGP to large-scale bipartite graphs, we further parallelize

Input: bipartite graph $\mathcal{G} = \langle \mathcal{U}_n \cup \mathcal{U}_f \cup \mathcal{A}, E, T \rangle$, lower bound of scores δ, error bound ϵ, exponent parameter β, regularization parameter α
Output: Scores of user nodes $\widehat{X}_{\mathcal{U}}$
for each $u \in \mathcal{U}_f$ **do**
  set $\widehat{x}_u = 1$.
end for
for each $u \in \mathcal{U}_n$ **do**
  set $(x_l, x_r) = (\delta, 1), x_m = (x_l + x_r)/2$
  if $\mathcal{L}(x_l) \geq \mathcal{L}(x_r)$ **then**
    while $\mathcal{L}(x_m) \geq \mathcal{L}(x_r)$ and $x_r - x_m > \epsilon$ **do**
      $x_l = x_m, x_m = (x_l + x_r)/2$
    end while
  else
    while $\mathcal{L}(x_m) \geq \mathcal{L}(x_l)$ and $x_m - x_l > \epsilon$ **do**
      $x_r = x_m, x_m = (x_l + x_r)/2$
    end while
  end if
  if $x_r - x_m \leq \epsilon$ or $x_m - x_l \leq \epsilon$ **then**
    $\widehat{x}_u = x_m$
  else
    $\widehat{x}_u = \text{GSM}(\mathcal{L}_u)$ on initial points (x_l, x_m, x_r)
  end if
end for
return $\widehat{X}_{\mathcal{U}}$

Algorithm 1: Computing the fraud scores of user nodes.

Input: bipartite graph $\mathcal{G} = \langle \mathcal{U}_n \cup \mathcal{U}_f \cup \mathcal{A}, E, T \cup W \rangle$, lower bound of scores δ, error bound ϵ, exponent parameter β, regularization parameter α, maximum iteration limit r_m
Output: ranking of $\widehat{X}_{\mathcal{A}}$
set $r = 1, \widehat{X}_{\mathcal{U}}^{(r-1)} = 0$
set $\widehat{X}_{\mathcal{U}}^{(r)}$ = the return values $\widehat{X}_{\mathcal{U}}$ of Algorithm 1
while $|\widehat{X}_{\mathcal{U}}^{(r)} - \widehat{X}_{\mathcal{U}}^{(r-1)}| > \epsilon$ and $r < r_m$ **do**
  for each $a \in \mathcal{A}$ **do**
    compute $\widehat{x}_a$ for target node a as in Eq. (11)
  end for
  for each $u \in \mathcal{U}$ **do**
    compute $\widehat{x}_u$ for source node u as in Eq. (12)
  end for
  $r = r + 1$
end while
return the ranking result of $\widehat{X}_{\mathcal{A}}$

Algorithm 2: iBGP.

our method on Apache Spark cluster, which is a well-known memory-based parallel computation framework [20]. In order to maximize the degree of parallelism, the parallel operators are mainly focused on the user dimension, since population of users is usually several orders of magnitude larger than that of the apps in real-world conditions.

In the parallel version of iBGP, each $u \in \mathcal{U}$ is defined as a key-value tuple $(u, \widehat{x}_u : \{(a, t_{ua} : w_{ua}) \mid a \in \mathcal{O}(u)\})$, where ":" is field delimiter within value. Step (4) in Algorithm 2 is computed in parallel directly. In steps (6)–(8), we first use a *map operator* on each $u \in \mathcal{U}$, so that the key-value tuple is transformed into $\{(a, w_{ua}\widehat{x}_u : w_{ua}) \mid a \in \mathcal{O}(u)\}$, and then a *reduce operator* is used to compute $\widehat{X}_{\mathcal{A}}$ as in (11); Steps (9)–(11) are calculated parallelly by simply utilizing a *map operator* on each user to sum up the weighted scores of related apps as in (12). Notice that the key-value tuples for users can be cached in memory to further improve efficiency.

4. Experiments

In this section, we perform experimental evaluation of iBGP and the competing methods on synthetic data, simulating click fraud, and real-world data from a mobile advertising platform. All of the algorithms are implemented on Apache

Table 1: Basic settings of synthetic data generation algorithm.

Type	Symbol	Description
Unlabeled users	U_n	Each unlabeled user randomly picks 1 to k_n apps as targets, with probability $1/(i^\lambda + C_1)$ to operate the ith app in app partite.
Fraud users	U_f	Each fraud user randomly picks 1 to k_f apps as targets, with probability p to operate fraud app. Particularly, $k_f \leq k_n$ and $p > \lvert A_f \rvert / \lvert A \rvert$.
Normal apps	A_n	The normal label is randomly assigned.
Fraud apps	A_f	The fraud label is randomly assigned.

Spark cluster [20] with six compute nodes (4 G RAM per node) and the raw data are stored on Hadoop Distribute File System (HDFS).

4.1. Synthetic Data. We first generate random user-app bipartite graphs with two sets of nodes, namely, app nodes A and user nodes U. In order to form a power law distribution on apps, we catch the intuition that apps are preliminarily sorted in descending order by their potential popularity. Basic settings of synthetic data generation algorithm are described in Table 1.

Note that the regularization constant C_1 can be derived by the cumulative probability $\sum_{i=1}^{|A|}(1/(i^\lambda + C_1)) = 1$.

According to our investigation on real-world data, population of users outnumbers that of the apps in most of the cases. We first simulate 3M unlabeled user nodes and 30K normal app nodes. Then, we injected 30K fraud users and 3K fraud apps into the bipartite graph and uniformly assigned the index i of app a_i. To evaluate the performance, we vary the following properties of the synthetic data.

Camouflage. As the injected fraudsters may try to mimic legitimate traffic to counter the detection methods, for example, operating more normal apps to diffuse fraudulent traffic, we scale the parameter p from 0.2 to 1 with interval equal to 0.2 to manipulate the camouflage level so that five different datasets are built with the global parameter settings: $\lambda = 1.5$, $C_1 = 12.14$, $k_n = 7$, and $k_f = 5$. We plot the app ID versus frequency of synthetic graphs with different values of p in Figure 3, where m in title "SG + m% R" denotes the strength of camouflage. After comparison, we notice that when the level of camouflage is small ($p \geq 0.6$), the majority of injected fraud apps lie on the upper side of normal ones, which can be easily caught since they have anomalously high frequencies than their counterparts. For $p < 0.6$, camouflage can hide the injected fraud apps in the dominating parts, proving challenge for detection.

Size of Fraud Users. We also set $p = 0.6$ and scale the injected fraudsters down to half of the size, namely, SG + 40% R^-, to test the robustness of our method.

Competing Algorithms. We carefully implement the popular graph propagation methods and their adaptations as competing algorithms: (a) NodeProp [8]: it is a propagation method based on a seed set of cheating source nodes. Only the scores of non-seed nodes are updated during the iteration; (b) EdgeProp [4]: it is a propagation method based on the agreeing/disagreeing dependencies between nodes, where a priori dependencies between nodes are needed; (c) HITS-o: we adopt the original HITS algorithm [13] based on weighted bipartite graphs to propagate fraud scores among users and apps; (d) BP-a: it is the propagation stage of the proposed algorithm in [16], where the Belief Propagation algorithm is adapted to incorporate background information. The likelihood of states of nodes are updated iteratively. We assign the fraud users as seeds in our competing algorithms, and all links associated with the seeds are labeled as disagreeing in EdgeProp. For BP-a, initial states distribution $\phi = \{P_n, P_f\}$ is assigned to each node, where subscript n denotes *normal* and f denotes *fraud*. The state distribution for seeds and nonseeds is $\phi_f = \{\delta, 1-\delta\}$ and $\phi_n = \{1-\delta, \delta\}$, respectively. The propagation matrix ψ is set as $\psi_{ii} = 1 - \delta$ and $\psi_{ij} = \delta$ $(i, j = 1, 2;\ i \neq j)$.

Evaluation. If we define the fraud apps by positive samples and the other apps by negatives, we can record the True Negative (TN), True Positive (TP), False Negative (FN), and False Positive (FP) rates to compute the popular metrics: precision, recall, and Cohen's Kappa statistic [21]. The parameter settings of iBGP are $\delta = 0.05$, $\beta = 2.1$, and $\alpha = 0.3$.

Table 2 shows the Kappa value on detecting fraud apps in the six synthetic bipartite graphs. For different levels of camouflage, iBGP consistently maintains the highest value. The gaps are increasingly obvious as the random property grows. BP-a algorithm tends to be much more sensitive to the variation of experimental settings than its counterparts, especially when the size of fraud users declines. Figure 4 plots the precision-recall curves of each method, where iBGP keeps its accuracy and constantly stays on top.

4.2. Real-World Data. We applied our method to advertisement logs from one of the mobile advertising platforms in China. Details of our datasets are formerly described in Section 3.2, which consists of seven days with around 2M users and 3.5K apps per day. Before entering the graph constructing stage, we first filter out the inactive apps based on their popularity. We choose a lower threshold $\theta_L = 2$ and remove the apps with users less than θ_L, such that approximately 2M users and 2K apps per day are eventually adopted to build the bipartite graph. Note that user pertinence is time-sensitive; in order to maintain representative user pertinence, a proper period of observing time is preferred. We partition our logs into seven subsets with a one-day period and conduct experiments on each subset to complete the evaluation.

Table 2: iBGP continuously maintains the highest Kappa value, despite various levels of camouflage or size of fraud users.

Graphs	iBGP	NodeProp	EdgeProp	Hits-o	BP-a
SG + 0% R	**1**	0.9620	0.9996	0.9624	0.7667
SG + 20% R	**0.9598**	0.9298	0.9489	0.9305	0.7469
SG + 40% R	**0.9342**	0.8976	0.9153	0.9002	0.6875
SG + 60% R	**0.8894**	0.8402	0.8386	0.8390	0.5369
SG + 80% R	**0.8366**	0.7432	0.8060	0.7775	0.4825
SG + 40% R^{-}	**0.9327**	0.8991	0.9252	0.9021	0.1833

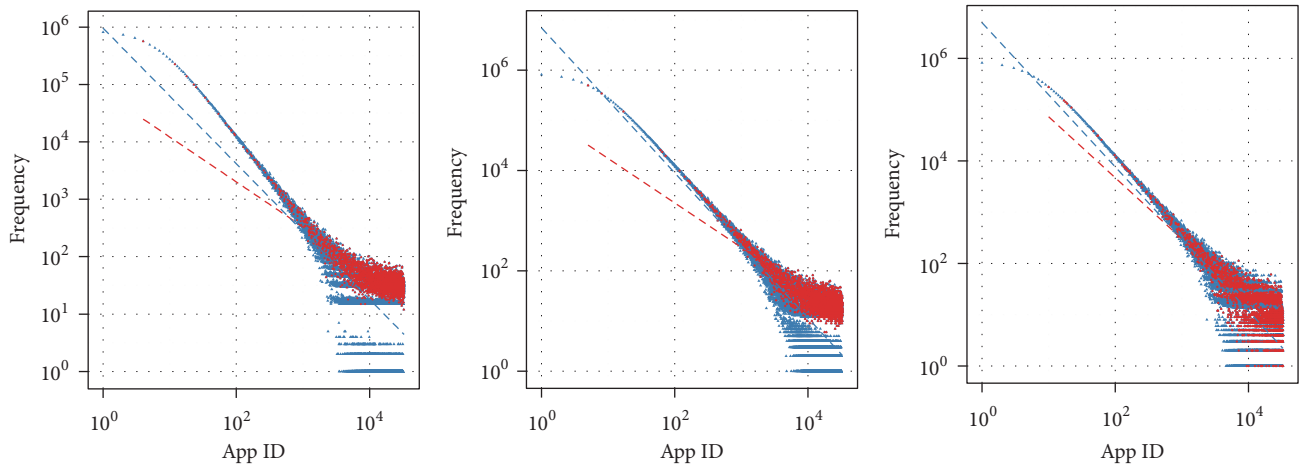


Figure 3: Synthetic graphs built with various degrees of camouflage. As we scale parameter p into smaller values, camouflage can hide the injected fraud apps in or put them close to dominating parts. Differences between the two types of apps are gradually disappearing as shown by the regression curves.

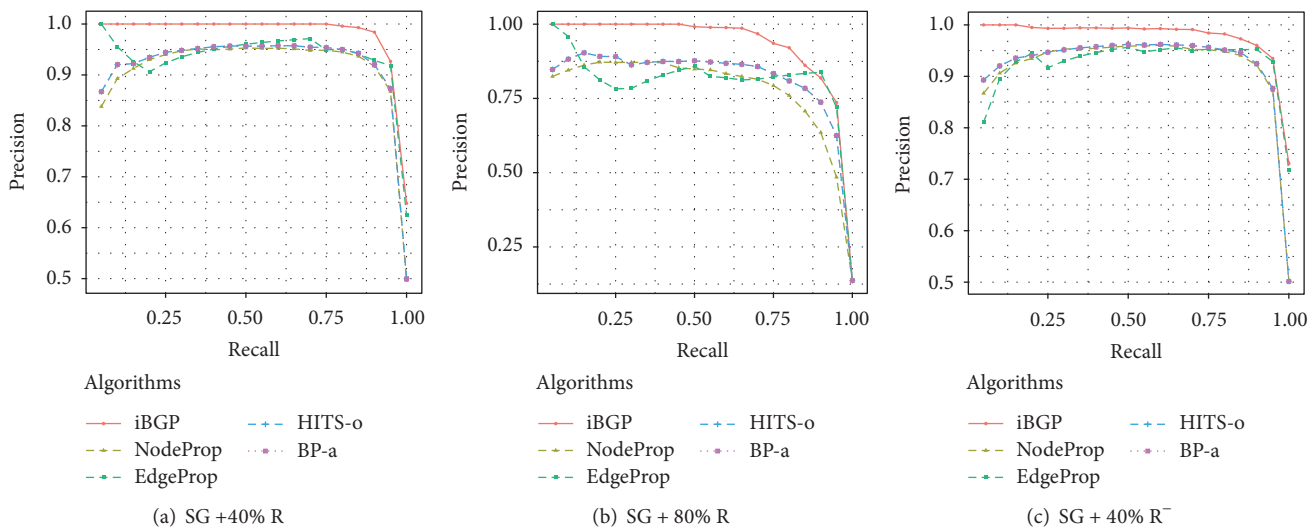


Figure 4: Precision versus recall curves of competing algorithms. iBGP reaches the highest precision and recall.

Table 3: iBGP outperforms each part in the seven-day real-world mobile ad dataset in terms of AUC value.

Date	iBGP	NodeProp	EdgeProp	HITS-o	BP-a
1st June	**0.9362**	0.5265	0.7671	0.7723	0.7629
2nd June	**0.9278**	0.5477	0.8217	0.6551	0.7758
3rd June	**0.9191**	0.5152	0.7897	0.7250	0.7292
4th June	**0.9118**	0.5091	0.7982	0.7754	0.7309
5th June	**0.8870**	0.4759	0.7952	0.6457	0.7249
6th June	**0.8719**	0.5385	0.8041	0.7041	0.7289
7th June	**0.8806**	0.5120	0.7799	0.7040	0.7069

Competing Algorithms. We have all the competing algorithms as in our former experiments.

Evaluation. We applied the aforementioned methods to each subset of logs and the output scores are sorted in descending order. For each method, the top-50 results are chosen to construct a candidate set, so that approximately 160 apps are sampled daily. Then, a manual labeling task based on empirical study is performed. The labeling process mainly follows the rules below:

(i) *Click and Install Profiles: Variation of Click Rate and Install Rate*. For example, apps whose hourly click rate bursts suddenly or consistently exceeds 10% on the observing day are highly suspicious.

(ii) *Geographical Distribution of Users*. Users that densely concentrate on a few geographically proximate cities are deemed to be abnormal.

(iii) *Mobility Characteristics of Users*. Device IDs or other users' mobile attributes are also studied. For example, users that share the same IP but with varied device IDs are considered to be generated by malwares.

We present the list of candidate apps to three domain experts, and a "fraud" label is given if at least two of the experts believe the app is fraudulent. Similar to the evaluation approach on synthetic data, we use precision, recall, and AUC (area under roc curves) to evaluate the effectiveness. The parameter settings of iBGP are $\delta = 0.05$, $\beta = 2.1$, and $\alpha = 0.3$.

Table 3 shows the AUC value on detecting the fraud-labeled apps from seven one-day period user-app bipartite graphs. Also in Figure 5 we plot the precision-recall curves of all algorithms. We analyze and explain the results as follows.

iBGP Outperforms HITS-o. The propagating approach of iBGP is identical with that of HITS-o. However, prior to propagation, iBGP learns the initial scores of users that HITS-o cannot capture. Experimental results in Figure 5 show that the initial scores of users do cause significant positive impact on the outcome.

iBGP Outperforms Other Methods. Results in Figure 5 and Table 3 demonstrate that iBGP continuously achieves higher performance. The intuitive insight of the results indicates that the quality gain from a well-considered initial score distribution of nodes may dramatically outweigh the marginal improvement induced by a more complicated propagation model.

4.3. Properties of iBGP

Accuracy on Top-Ranking Apps. Experiments on both synthetic data and real-world data shows that, for recall <0.2, iBGP keeps its precision close to 1, which means that the ranking order of iBGP could be more informative than that of the competing algorithms. In order to explore the reason, we set $\alpha = 0$ to exclude the effect of power law, leaving only the user pertinence model to determine the initial user scores (named iBGP-0). Results are shown in Figure 6, where iBGP-0 outperforms HITS-o weakly in most of the cases but consistently lies below the curve of iBGP, indicating that user pertinence is both informative and noisy. It indeed improves the model performance; yet high accuracy on top-ranking apps is, actually, mostly driven by the constraint of power law distribution on user scores.

Setting of β. Empirical study on real-world network [22] points out that the scaling parameter of power law distribution typically lies in the range $2 \leq \beta \leq 3$. We perform linear regression on all the user features in log-log plots (see, e.g., Figure 2) and work out the mean scaling parameter as $\overline{\beta} = 2.1$, which is β setting in our experiments.

Robustness with respect to α. We evaluate the performance on different values of the significant factor of power law; the algorithm with particular value of α is named as iBGP-α. Result in Figure 6 shows that the performance of iBGP is stable, where relatively weak sensitivity to changes in α is obtained under the range $\alpha \leq 0.3$. However, overstrengthening of power law could offset the effect on the model of user pertinence. We suggest setting $\alpha \leq 0.3$ for other unseen circumstances.

5. Conclusion

We analyze the fraud detection problem in mobile advertising to detect fraudulent apps and introduce the initial score learning model to a large user-app bipartite graph propagation method for fraud detection. With the careful investigation of behavior patterns of mobile app users, we identify two key characteristics: power law distribution and user pertinence. We mathematically formulate the two findings and propose a new propagation method on bipartite graph called iBGP. In contrast to the traditional methods that often explicitly or potentially assign a certain value as initial scores for non-seed nodes, the core step before user score propagation of our

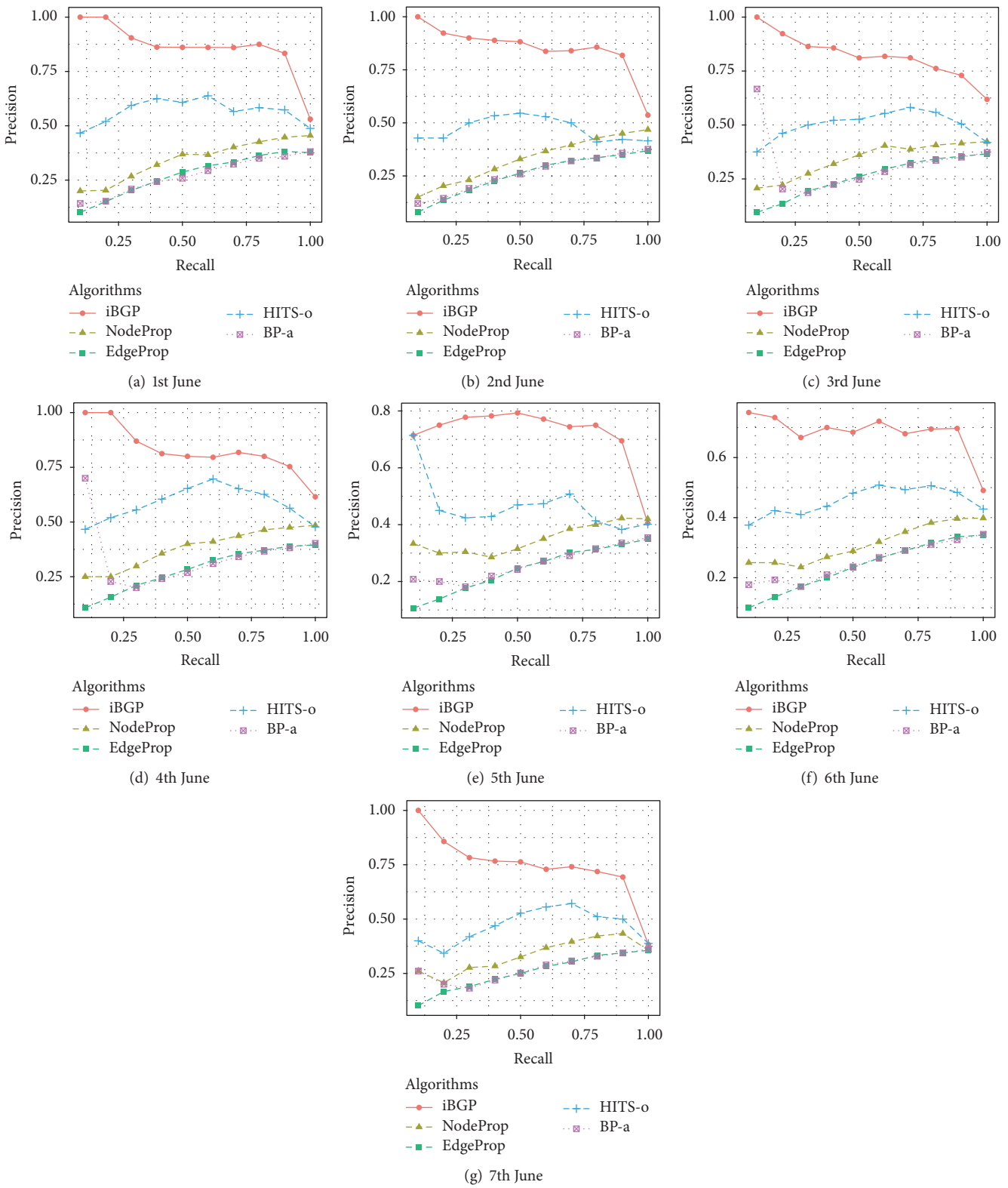
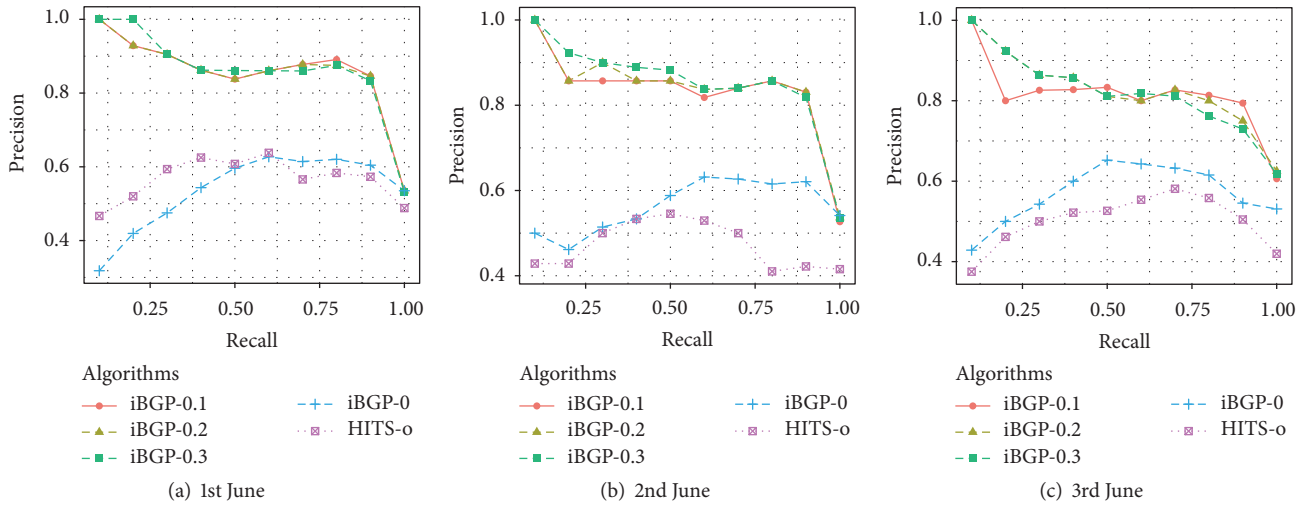


Figure 5: iBGP achieves higher precision and recall in the seven-day real-world mobile ad dataset.



(a) 1st June (b) 2nd June (c) 3rd June

FIGURE 6: iBGP with different settings of α. iBGP-α ($0 < \alpha \leq 0.3$) significantly outperforms iBGP-0, elucidating that power law is crucial for iBGP, especially for the accuracy on top-ranking apps. iBGP is robust to changes in α.

model is to learn the initial scores of non-seed users based on the user behavior patterns. Our method is intrinsically parallelizable, and experimental results demonstrate that we effectively detect fraudulent apps with high accuracy especially for the top-ranking ones, which is superior to popular traditional graph propagation methods and their adaptations.

Symbols and Definitions

$\mathcal{U} = \mathcal{U}_n \cup \mathcal{U}_f$: $\mathcal{U}$ is the set of user nodes, $\mathcal{U}_n$ is the normal set, and $\mathcal{U}_f$ is the fraud set

$\mathcal{A} = \mathcal{A}_n \cup \mathcal{A}_f$: $\mathcal{A}$ is the set of app nodes, $\mathcal{A}_n$ is the normal set, and $\mathcal{A}_f$ is the fraud set

$\mathcal{O}(u)$: The set of node u's targets

$\mathcal{I}(a)$: The set of node a's sources

x_i: The fraud score of node i

V_{ua}: The subset of behavior logs that contain u and a

T_{ua}: The subset of behavior logs that u targets at a

w_{ua}: The weight of e_{ua}.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Science and Technology Planning Project of Guangdong Province, China (no. 2013B090500087, no. 2014B010112006), the Scientific Research Joint Funds of Ministry of Education of China and China Mobile (no. MCM20150512), and the State Scholarship Fund of China Scholarship Council (no. 201606155088).

References

[1] A. Zarras, A. Kapravelos, G. Stringhini, T. Holz, C. Kruegel, and G. Vigna, "The dark alleys of madison avenue: understanding malicious advertisements," in *Proceedings of the ACM Internet Measurement Conference (IMC '14)*, pp. 373–379, Vancouver, Canada, November 2014.

[2] A. Metwally, D. Agrawal, and A. E. Abbadi, "Using association rules for fraud detection in web advertising networks," in *Proceedings of the 31st International Conference on Very Large Data Bases*, pp. 169–180, VLDB Endowment, August-September 2005.

[3] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[4] H. Dai, F. Zhu, E.-P. Lim, and H. H. Pang, "Detecting anomalies in bipartite graphs with mutual dependency principles," in *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM '12)*, pp. 171–180, IEEE, Brussels, Belgium, December 2012.

[5] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen, "Combating web spam with trustrank," in *Proceedings of the 30th International Conference on Very Large Data Bases*, vol. 30, pp. 576–587, VLDB Endowment, 2004.

[6] V. Krishnan and R. Raj, "Web spam detection with antitrust rank," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (AIRWeb '06)*, vol. 6, pp. 37–40, Seattle, Wash, USA, August 2006.

[7] X. Li, Y. Liu, M. Zhang, and S. Ma, "Fraudulent support telephone number identification based on co-occurrence information on the web," in *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pp. 108–114, July 2014.

[8] X. Li, M. Zhang, Y. Liu, S. Ma, Y. Jin, and L. Ru, "Search engine click spam detection based on bipartite graph propagation," in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*, pp. 93–102, ACM, February 2014.

[9] X. Zhang, Y. Wang, N. Mou, and W. Liang, "Propagating both trust and distrust with target differentiation for combating link-based Web spam," *ACM Transactions on the Web*, vol. 8, no. 3, article 15, 2014.

[10] M. Agosti and L. Pretto, "A theoretical study of a generalized version of Kleinberg's HITS algorithm," *Information Retrieval*, vol. 8, no. 2, pp. 219–243, 2005.

[11] B. Kitts, J. Y. Zhang, G. Wu et al., "Click fraud detection: adversarial pattern recognition over 5 years at microsoft," in *Real World Data Mining Applications*, vol. 17 of *Annals of Information Systems*, pp. 181–201, Springer International, Cham, Switzerland, 2015.

[12] B. Wu, V. Goel, and B. D. Davison, "Topical TrustRank: using topicality to combat web spam," in *Proceedings of the 15th International Conference on World Wide Web (WWW '06)*, pp. 63–72, ACM, May 2006.

[13] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, vol. 46, no. 5, pp. 604–632, 1999.

[14] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, "Netprobe: a fast and scalable system for fraud detection in online auction networks," in *Proceedings of the International Conference on World Wide Web (WWW '07)*, pp. 201–210, ACM, Alberta, Canada, May 2007.

[15] D. Koutra, T. Y. Ke, U. Kang, D. H. Chau, H. K. K. Pao, and C. Faloutsos, "Unifying guilt-by-association approaches: theorems and fast algorithms," *Lecture Notes in Computer Science*, vol. 6912, no. 1, pp. 245–260, 2011.

[16] A. Tamersoy, K. Roundy, and D. H. Chau, "Guilt by association: large scale malware detection by mining file-relation graphs," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*, pp. 1524–1533, August 2014.

[17] D. Chakrabarti, S. Funiak, J. Chang, and S. A. Macskassy, "Joint inference of multiple label types in large networks," in *Proceedings of the 31st International Conference on International Conference on Machine Learning (ICML'14)*, Beijing, China, June 2014.

[18] P. P. Talukdar and K. Crammer, "New regularized algorithms for transductive learning," in *Machine Learning and Knowledge Discovery in Databases*, pp. 442–457, Springer, 2009.

[19] M. Minoux, *Mathematical Programming: Theory and Algorithms*, John Wiley & Sons, 1986.

[20] A. G. Shoro and T. R. Soomro, "Big data analysis: apache spark perspective," *Global Journal of Computer Science and Technology*, vol. 15, no. 1, 2015.

[21] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[22] A. Clauset, C. R. Shalizi, and M. E. Newman, "Power-law distributions in empirical data," *SIAM Review*, vol. 51, no. 4, pp. 661–703, 2009.