HGsuspector: Scalable Collective Fraud Detection in Heterogeneous Graphs

Xiang Li
JD Finance
Beijing, China
lixiang29@jd.com

Wen Zhang
JD Finance
Beijing, China
zhangwen11@jd.com

Jiuzhou Xi
JD Finance
Beijing, China
xijiuzhou@jd.com

Hao Zhu
JD Finance
Beijing, China
zhuhao6@jd.com

ABSTRACT

Graph can straightforwardly represent the relations between the objects, which inevitably draws a lot of attention of both academia and industry. Achievements mainly concentrate on homogeneous graph and bipartite graph. However, it is difficult to use existing algorithm in actual scenarios. Because in the real world, the type of the objects and the relations are diverse and the amount of the data can be very huge. Considering of the characteristics of "black market", we propose *HGsuspector*, a novel and scalable algorithm for detecting collective fraud in directed heterogeneous graphs. We first decompose directed heterogeneous graphs into a set of bipartite graphs, then we define a metric on each connected bipartite graph and calculate scores of it, which fuse the structure information and event probability. The threshold for distinguishing between normal and abnormal can be obtained by statistic or other anomaly detection algorithms in scores space. We also provide a technical solution for fraud detection in e-commerce scenario, which has been successfully applied in Jingdong e-commerce platform to detect collective fraud in real time. The experiments on real-world datasets, which has billion nodes and edges, demonstrate that HGsuspector is more accurate and fast than the most practical and state-of-the-art approach by far.

KEYWORDS

Bipartite Graph; Heterogeneous Graph; Connected Bipartite Subgraph; Edge Density Function; Fraudulent Behavior;

ACM Reference Format:
Xiang Li, Wen Zhang, Jiuzhou Xi, and Hao Zhu. 2018. HGsuspector: Scalable Collective Fraud Detection in Heterogeneous Graphs. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

With the rise of e-commerce, both the platforms and their users benefit from the great change in business model. However, it introduces a series of frauds and causes serious risks at the same time. Malicious users cooperate with each other and commit collective fraud in the platform for illegal profits. With a clear division of labor, they eventually form steady organizations and cause great damage to the e-commerce platforms and normal users. Thus a series of algorithms are proposed to detect the collective frauds, such as fake review and fake rate, in recent years.

Traditional approaches of fraud detection mainly relied on auditing, which is inefficient and unreliable [25]. However, machine intelligence (MI)-based techniques employing conventional data mining algorithms have been proven to be useful for detecting anomalies, especially in large data sets. With the promotion of graph-based data, graph anomaly detection attracts many attentions of researchers in recent years rapidly. Compared with the conventional techniques, there are several reasons it might be preferable to use graphs for anomaly detection, as discussed in [2]. Due to the challenges of traditional methods investigated in survey [2], graph-based anomaly detection could effectively find abnormal structures with (semi-)unsupervised techniques.

However, most of the previous algorithms only focus on fraud in a certain scenario and don't give a systematical solution within a real e-commerce situation with coherent scenarios. With the analyses of various kinds of fraud cases among all the scenarios on our e-commerce platform, we find that based on the modularization and process standardization of operation, the "black market" can produce a number of fake account in a short period of time, but they also leave their traces in the patterns of the information generation and authenticity improvement. Based on the prior knowledges, we provide a novel and effective method *HGsuspector* to improve the performance of collective fraud detection on heterogeneous graph which has been successful used on our e-commerce platform.

In summary, the main contributions of this paper are summarized as follows:

- **Feasibility of reducible heterogeneous graphs:** for any heterogeneous graph, we could always decompose it into irreducible bipartite graphs on which we can detect fraudulent behaviors in the specific scenario, according to the relation types of the problem domain.

- **Metric on connected bipartite graph:** a metric is proposed to reveal the suspicious level of connected bipartite graph preserving both structural information and node/edge attributes.
- **Edge density function:** we propose a density function $p(i, j)$ for enhancing the ability of detecting suspicious subgraphs and list several edge density functions. For each edge density function, it corresponds to a fraud pattern to be detected which depends on the problem domain.
- **Efficiency and Scalability:** *HGsuspector* is very efficient in calculating scores on connected bipartite subgraphs and could process large graphs with billion nodes and edges.

The remaining of this paper is organized as follows. Section 2 states recent work on graph-based anomaly detection algorithms. In section 3, we formally define our problem, propose a metric on connected bipartite subgraphs, and present our algorithm *HGsuspector*. Experiments are conducted to demonstrate the efficiency and scalability of *HGsuspector* in section 4, compared with other state-of-art algorithms. Finally, we draw conclusions and discuss some related work in section 5.

2 RELATED WORK

Over the past few decades, graph-based anomaly detection algorithms have developed vastly with the explosion of big data. These methods could be applied to both the static and dynamic graphs with labels or attributes, and cover lots of areas of networks such as security, health-care, financial networks and so on. Graph-based anomaly detection could be categorized into two classes: quantitative detection and qualitative explanation [2]. In applications, those two methods are both used to detect fraud precisely from structural information or from attributes of nodes and edges of networks. Meanwhile, algorithms of graph-based semi-supervised learning (GSSL) [17, 20, 26] predict the label of the unlabelled nodes with some manually labeled seed risky nodes.

Anomaly detection with graph embedding is a new approach emerging in recent years. These methods either have their own semantic embeddings according to the characteristics of the graph [1, 6, 7], or embed the whole graph with existing embedding algorithms and then detect anomalies with traditional machine learning algorithms. Existing graph embedding methods can be categorized into mainly three classes: factorization [5, 16], random walk [10, 19, 21], deep learning [13, 24] and others [23] according to the survey [9]. After a graph is embedded into a p-dimensional vector space, then the embedding can be used to cluster the nodes. Graph clustering can be of two types [21]: (a) structure-based, (b) attribute based clustering. For structure based clustering, it aims to identify dense subgraphs or nodes with similar roles (outliers). Hence, embedding based anomaly detection methods are of 2-step: embedding and clustering.

The embedding method mentioned above is on the basis of the feature-based approaches [2], and they need either large computation or feature engineering on graphs, which can not scale to large graphs. To overcome this problem, metric-based methods introduce a metric on a subgraph to quantify the normality of the subgraph [11, 12, 14, 18]. *Fraudar* [11] detects fraudulent blocks by introducing a density metric to measure the total suspiciousness of the gradually reduced bipartite graph. *HoloScope* [14], on the basis of Fraudar, introduces a dynamic weighting approach to detect not only dense fraudulent blocks but also low-density ones. In addition, it can also be adapted to the sudden bursts and drops of attacking patterns. *AMEN* [18] detects anomalous neighborhoods with a quality score *normality* that utilizes both structure and attribute information. *CatchSync* [12] can automatically detect suspicious connectivity patterns from a directed graph with a parameter-free algorithm, using *synchronicity* and *normality* to measure the behavior pattern of the bipartite graph. Among these methods, *FRAUDAR*, *HoloScope* and *AMEN* define the metric on the subgraph, while *CatchSync* is on the node. For the purpose of detecting suspicious subgraphs, the previous three methods are unable to discriminate two directed subgraphs with the same undirected structure.

3 METHODOLOGY

In this section, we will present *HGsuspector*, a novel and effective algorithm for detecting collective fraud. It mainly focuses on defining problem, introducing our metric and finally proposing our algorithm *HGsuspector*. Firstly we introduce several symbols and terminologies that will be used throughout the whole paper and define the problem we aim to solve. Next, we propose a novel metric on connected bipartite graphs and discuss several properties of this metric. Finally, we propose our algorithm for detecting suspicious subgraphs on the basis of the metric.

3.1 Definition of collective fraud on e-commerce platform

As the mediation between shops and users, the e-commerce platform would have promotion activities at intervals to increases sales and profits. However, the fraudsters saw the great opportunity in promotion activities at the same time. As the preferential measure is always subjected to the account, the fraudsters may operate many accounts in a certain time range for illegal profits during each promotion, which lead to a character convergence of these abnormal accounts. We define this kind of fraud in e-commerce platform as collective fraud. The typical scenarios of collective fraud include shop brushing for reputation increment, bonus-getting and discount scalper in promotion activities, money laundering by transaction, etc.

Most of these accounts are provided by "black market" and registered automatically by programs. To evade the fraud detection, the fraudsters frequently change the IP, the MAC, the phone number and other information by a series technique in variable scenarios. Even more, the accounts will be camouflaged by simulation operations like login and even browsing at intervals randomly. All above operations make the abnormal accounts look like normal. We call this processing "raising account".

Figure 1 shows an example of mainly scenarios of an account which involves variable node types and relation types. With the analyses and comparison of a large number of normal accounts and abnormal accounts among all the scenarios on our e-commerce platform, we find some clues. The random strategies of information generation make the false information independent of each other

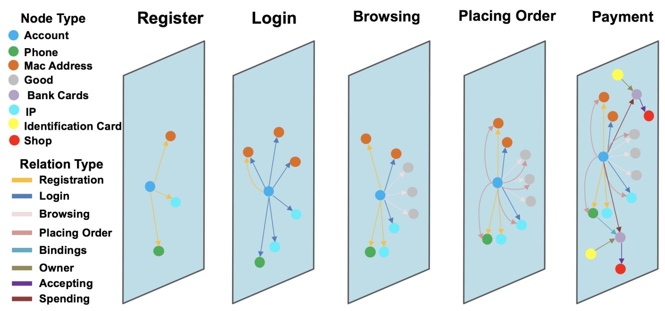


Figure 1: An example of mainly scenarios of an account which involves variable node types and relation types.

Table 1: Symbols of graph terminologies

Symbols	Interpretation
V	vertices(nodes) set of a graph
E	edges(relations) set of a graph
G	graph $G = (V, E)$
B	bipartite graph $B \subset G$
S	subject set of B
O	object set of B
A_{ij}	adjacency matrix
a_i	the $i-th$ node in A_{ij}
a_{ij}	element of A_{ij}
e_{ij}	edge connecting node a_i and a_j
d_i	degree of node a_i
$p(i, j)$	density function of edge e_{ij}
S^s_{ij}, S^d_{ij}	score of an edge e_{ij}
S_s, S_d	score of a connected bipartite graph
$\mathcal{N} = \{N_1, N_2, \cdots, N_m\}$	set of node type
$\mathcal{R} = \{R_1, R_2, \cdots, R_n\}$	set of relation type
$\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_k\}$	set of application scenario.
$\mathcal{B}_r$	set of connected graph for relation $r \in \mathcal{R}$.
#cc	number of connected graphs.

in the same scenario. But the fraud behaviors, such as simulating normal browsing to popular goods in the processing of account masquerade, are consistent. It means we could consider about each type of false information related to the account of a scenario respectively, and then integrate the discrete results to judge collective fraud. Meanwhile, we could filter the abnormal account throughout all account life-cycle.

3.2 Problem Definition

Table 1 gives a complete list of the symbols we use throughout the paper. Now we formally give the definition of the directed heterogeneous graph (i.e. heterogeneous information network) that is similar to [22].

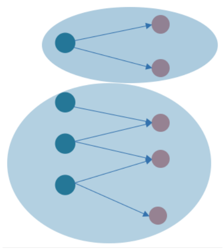


Figure 2: A directed bipartite graph which contains two connected directed bipartite graphs.

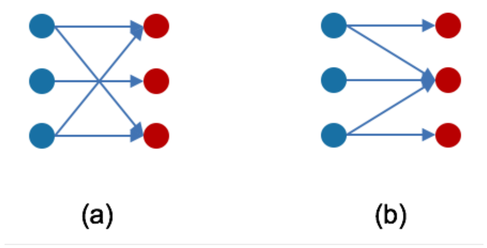


Figure 3: Symmetric and asymmetric bipartite graphs. (a) Symmetric bipartite graph. For each edge e_{ij}, it has the property that $d_i = d_j$, and it contains two connected components, each of which is a k-regular graph; (b) Asymmetric bipartite graph. For all edges in it, there always exists an edge that satisfies $d_i \neq d_j$.

Definition 3.1. *Directed Heterogeneous Graph*: A directed heterogeneous graph (DHG) is a graph $G = (V, E)$ with node type $n \in \mathcal{N}$ and relation type $r \in \mathcal{R}$, where $|\mathcal{N}| > 1$ or $|\mathcal{R}| > 1$.

Definition 3.2. *Directed Bipartite Graph and Connected Directed Bipartite Graph*: A directed bipartite graph (DBG) B is a directed graph, its vertices are composed of subject set S and object set O, where $S \cap O = \emptyset$. For an edge e_{ij}, it satisfies the condition that source node $a_i \in S$ and destination node $a_j \in O$. More specifically, a DBG is composed of connected directed bipartite graphs (CDBG), in which for arbitrary two nodes u and v there always exists a path connecting u and v after all edges in B are replaced with undirected edges. Especially, DBG is a bi-DHG composed of 2 node types and only one edge type. See Figure 2.

Definition 3.3. *Symmetric and Asymmetric Bipartite Graph*: A asymmetric bipartite graph B is a graph that always contains an edge $e_{ij} \in E$ satisfied that $d_i \neq d_j$, where d_i and d_j are the degree of vertex a_i and a_j respectively. If $d_i = d_j$ for all edges in B, then B is symmetric. In fact, symmetric bipartite graph is a regular bipartite graph (graph where every vertex has the same degree) containing multiple connected components, each of which is regular. See Figure 3.

Now we define our problem formally. As discussed in [4], we describe our problem as *collective fraud detection* that aims to detect both the suspicious nodes and the subgraphs within a heterogeneous graph in an unsupervised approach with structural and attribute information preserved. And furthermore verify the application scenario $s \in \mathcal{S}$ of each suspicious subgraph detected (e.g., register and login). It is defined as follows.

Given: a directed heterogeneous graph $G = (V, E)$ with $|V|$ nodes.

Define: a metric to evaluate the suspicious level of connected bipartite subgraph, similar to that in [12, 18].

Find: a set of connected bipartite subgraphs $\{\mathcal{B}_r\}$. The suspicious nodes can be further obtained from the union of nodes in $\{\mathcal{B}_r\}$.

3.3 Metric

In this section, we propose a novel metric on connected bipartite graphs to measures the suspiciousness of a connected bipartite graph. The novelty is that the proposed metric distinguishes the graphs with the same structure, which often represent different behavior patterns, to improve the accuracy of the detection. Furthermore, we also propose several edge density functions as the attribute component in the metric in order to enhance the ability to detect suspicious subgraphs.

In Newman's [15] work, modularity is introduced as follows.

$$M = \frac{1}{2m} \sum_{i,j} (a_{ij} - \frac{d_i d_j}{2m}), \tag{3.1}$$

It measures the strength to divide graphs into modules. Consequently, it is suitable to measure the clustering characteristics of connected subgraphs within a graph. According to the decomposition schema of a directed heterogeneous graph (See section 3.4), we score each connected bipartite subgraph. However, two connected subgraphs with the same structure using Equation 3.1 will yield the same score (See Figure 4), thus we need a technique to distinguish these connected subgraphs since they might stand for very different behaviors.

Based on this fact, we propose our metric on a connected bipartite graph which is defined as follows.

$$S_s = \sum_{i,j} \frac{a_{ij} - d_i d_j / 2m}{d_i} \cdot p(i,j) \tag{3.2}$$

$$S_d = \sum_{i,j} \frac{a_{ij} - d_i d_j / 2m}{d_j} \cdot p(i,j), \tag{3.3}$$

where m is the number of edges of a connected bipartite graph, S_s and S_d are scores of all possible edges normalized by the degree of each endpoint for each connected bipartite graph respectively. The item in S_s consists of two parts. The first part describes the structural information as modularity does, normalized by the degree of source node a_i. While the second part $p(i,j)$ reveals the strength node a_i and a_j co-occur, e.g., edge density function. thus it depends on the problem domain heavily. The item in S_d is the same as that in S_s.

Furthermore, $p(i,j)$ can be defined as follows:

$$p(i,j) = \begin{cases} g(i,j), & a_{ij} = 1 \\ 1, & a_{ij} = 0, \end{cases} \tag{3.4}$$

where $g(i,j)$ is the emerging probability of edge e_{ij} with endpoints type $A, B \in \mathcal{N}$ and $A \neq B$ when $a_{ij} = 1$. In the case of $a_{ij} = 0$, i.e., node a_i and a_j are not connected in the observed graph, we

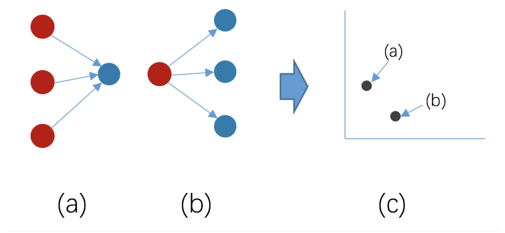


Figure 4: (a) and (b) are distinct bipartite graphs and they share the same structure, which *modularity* [15] can not distinguish. (c) is the mapped points of (a) and (b).

penalize the contribution of the score with only a structure item $-\frac{d_i d_j / 2m}{d_j} = -\frac{d_i}{2m}$. Hence, for each pair of nodes (a_i, a_j) doesn't form an edge in a connected bipartite graph, we give a penalty score $(-\frac{d_j}{2m}, -\frac{d_i}{2m})$ which corresponds to the normalization of source and destination nodes respectively.

According to the definitions above, we have the following lemmas.

LEMMA 3.1. *Score defined in Equations 3.2 and 3.3 can distinguish any two arbitrary asymmetric bipartite graph B_1 and B_2 with opposite directions. See Figure 4 for illustrating.*

Proof. *Denote the number of edges of B_1 and B_2 by m. For an edge e_{ij} in B_1, it corresponds to edge e_{ji} in B_2, and thus the degrees (d_i, d_j) of endpoints of e_{ij} correspond to (d_j, d_i) of edge e_{ji} in B_2. For edges e_{ij} and e_{ji}, their scores are*

$$S_{e_{ij}} = \frac{1 - \frac{d_i d_j}{2m}}{d_i}, S_{e_{ji}} = \frac{1 - \frac{d_j d_i}{2m}}{d_j} \tag{3.5}$$

respectively. Summing over on each side of the equations, we can obtain

$$\begin{aligned} S^e_{1,s} &= \sum_{e_{ij}} S_{e_{ij}} = \sum_{e_{ij}} \frac{1 - \frac{d_i d_j}{2m}}{d_i} = S^e_{2,d}, \\ S^e_{2,s} &= \sum_{e_{ji}} S_{e_{ji}} = \sum_{e_{ji}} \frac{1 - \frac{d_j d_i}{2m}}{d_j} = S^e_{1,d}, \end{aligned} \tag{3.6}$$

where the equations in 3.6 represent scores of edges in both B_1 and B_2. Similarly, we can obtain the penalties of C_1 and C_2, i.e.,

$$\begin{aligned} S^p_{1,s} &= S^e_{2,d}, \\ S^p_{2,s} &= S^e_{1,d}. \end{aligned} \tag{3.7}$$

Finally, we obtain

$$\begin{aligned} S_{1,s} &= S^e_{1,s} + S^p_{1,s} = S_{2,d}, \\ S_{1,d} &= S^e_{1,d} + S^p_{1,s} = S_{2,s}. \end{aligned} \tag{3.8}$$

For C_1 and C_2, their corresponding scores follows that

$$(S_{1,s}, S_{1,d}) = (S_{2,d}, S_{2,s}) \neq (S_{2,s}, S_{2,d}). \tag{3.9}$$

Thus the lemma follows. □

Lemma 3.1 indicates that for any two asymmetric bipartite graphs, we can always map two graphs into points on a plane if their euclidean distance not equal to zero, e.g., distinguishing the two connected subgraphs.

For complete bipartite graph, there will be $|S| * |O|$ edges with $|S|$ and $|O|$ vertices in S and O respectively without punishment. For each edge e_{ij}, the degrees of the endpoints satisfy that $d_i d_j = m$. Thus the corresponding score of the complete bipartite graph is $(\frac{1}{2d_i}, \frac{1}{2d_j})$, where $d_i \geq 1$ and $d_j \geq 1$. And we have the following lemma (or property):

Lemma 3.2. *For any complete bipartite graph B, the lower bound of score with edge density function $g(i,j) = 1$ satisfies $S_s \geq \frac{1}{2}$ and $S_d \geq \frac{1}{2}$. Furthermore, the score of B is $(\frac{|O|}{2}, \frac{|S|}{2})$.*

Proof. *From the definition of degree, we have $d_v \geq 1$ for arbitrary complete bipartite graph. Thus it easily follows that $S_s \geq \frac{1}{2}$ and $S_d \geq \frac{1}{2}$.*

For each vertex $a_i \in S$, we have $d_i = |O|$. Similarly, $d_j = |S|$. Thus we have

$$\sum_{a_i \in S} \frac{1}{2d_i} = \frac{m}{2d_i} = \frac{d_j}{2} = \frac{|O|}{2}, \tag{3.10}$$

$$\sum_{a_j \in O} \frac{1}{2d_j} = \frac{m}{2d_j} = \frac{d_i}{2} = \frac{|S|}{2}. \tag{3.11}$$

Hence, the lemma follows. □

Lemma 3.2 gives the lower bound of score on 2-dimensional vector space, and it is proportional to the number of edges in a complete bipartite graph. It is also indicated that the score on a complete bipartite graph are resting on integer coordinates.

To capture domain specific suspicious nodes, we need to enhance our score Equations 3.2 and 3.3. Therefore we propose several types of edge density functions $p(i,j)$, which is listed as follows:

- $p(i,j) = 1$: Any additional attribute information of nodes and edges is ignored, and the density is only related to its structure, namely, the distribution of the probability of all edges are uniform. In this case, the scores are more suitable to distinguish connected subgraph with different structures.
- $p(i,j) = \frac{\sum_{i \in S, j \in O} e_{ij}}{m}$. m is the number of edges of the whole directed heterogeneous graph, and the numerator is the total number of the edge type $r(s,o) \in \mathcal{R}$. It reveals the prior probability of the edge type $r(s,o)$, and is more apt to distinguish two connected subgraphs with the same structure but different distributions of edge types.
- $p(i,j) = P(e_{o_1 o_2 \cdots o_t}, r(s,o))$. $r(s,o) \in \mathcal{R}$ is the edge type where $s \in S$ and $o \in O$, and $e_{o_1 o_2 \cdots o_t}$ is the event sequences occurring with edge type $r(s,o)$. This edge density function describes the joint probability of $r(s,o)$ and $e_{o_1 o_2 \cdots o_t}$, and can reflect the pattern of rare events occurring between s and o. It is very useful to detect suspicious subgraphs rare events occurring.

3.4 Algorithm

The algorithm consists of two procedures: *HGsuspector* and *CalculateScore*. The former inputs a directed heterogeneous graph and outputs a set of suspicious bipartite subgraphs. The latter calculate scores of a given connected bipartite graph as detailed in algorithm 1.

Algorithm 1: HGsuspector(G) // detects suspicious structures in a large DHG.

```
Input: a DHG $G$.
Output: suspicious CDBG.
1  decompose $G$ into $k$ directed bipartite graphs
     $\mathcal{B} = \{B_1, B_2, \cdots, B_k\}$, where $k = |R|$
2  $\mathcal{S} \leftarrow \{\}$    // suspicious set of bipartite graph of
     each type.
3
4  foreach $B$ in $\mathcal{B}$ do
5      $C \leftarrow ConnectedComponents(B)$
6      $S_B \leftarrow \{\}$      // score set of bipartite graph $B$
7
8      foreach $C$ in $\mathcal{C}$ do
9          $s \leftarrow CalculateScore(C)$
10         add $s$ to $S_B$
11     $\hat{S_B} \leftarrow DetectOutlier(S_B)$
12     add $\hat{S_B}$ to $\mathcal{S}$
13 Return $\mathcal{S}$
```

Algorithm 2: CalculateScore(C)

```
Result: a 2-dimensional score vector of the given connected
        bipartite subgraph.
1  $(S_s, S_d) \leftarrow$ calculate scores of $C$ with Equations 3.2 and 3.3. ;
2  Return $(S_s, S_d)$
```

The algorithm 1 first decomposes the whole DHG into k directed bipartite graphs. The number of bipartite graphs obtained depends on the size of $\mathcal{R}$ as illustrated in Table 1. For each relation type $r \in \mathcal{R}$, it corresponds to a bipartite graph in G, thus G can always be decomposed into k edge-disjoint bipartite graphs.

Of the main procedure *HGsuspector*, *ConnectedComponents*(B) is the subprocedure calculating the connected components of the given bipartite graph B, which yields a set of connected bipartite subgraphs denoted by $\mathcal{C}$. And then *DetectOutlier* detects outliers in the graph with scores.It can be any outlier detection algorithms (e.g., *dbscan* and so on).

Finally, we obtain the suspicious subgraphs and further the suspicious nodes within these subgraphs.

4 EXPERIMENTS

In this section, we conduct the experiments with *HGsuspector* and several other fraud detection algorithms to demonstrate its effectiveness and scalability on our real-world dataset. Finally, we demonstrate that *HGsuspector* could outperform other state-of-the-art algorithms.

4.1 Experiment setup

We utilize a Decomposition-Score (DS) step to conduct our experiments. At first, we decompose a DHG into multiple bipartite graphs on a real-world dataset, score each connected bipartite subgraph,

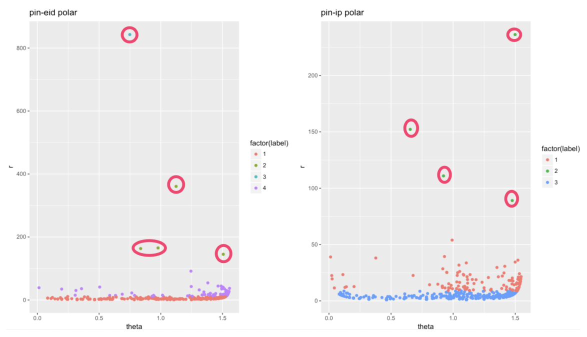


Figure 5: Scatter plot of scores of *pin-ip* and *pin-eid* bipartite graph in login stage after transforming the axis into polar coordination, and each point corresponds to a specific connected bipartite subgraph. The points circled in red are the abnormal structures detected.

Table 2: Summary of DHG dataset from our e-Commerce platform.

#user	#device id(eid)	#IP	total
77.5M	101.4M	36.8M	215.7M

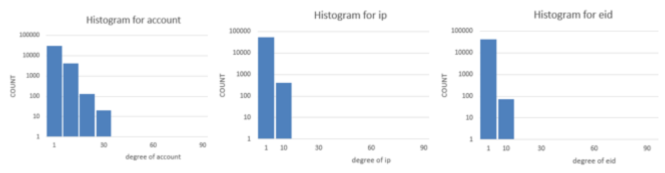


Figure 6: Histogram for user, ip and eid in the heterogeneous graph in login scenario. It is apparent that all these histograms follow a power law distribution.

and find the suspicious subgraphs with anomaly detection algorithms. Next, we extract the nodes from the suspicious subgraphs detected and calculate the precision and recall. Finally, we run other algorithms on the decomposed bipartite graphs to obtain the suspicious nodes and compare the corresponding performance with *HGsuspector*.

4.2 Data set

Here we provide a dataset describing users' login behaviors within one month(Jun 2017) collected from our e-Commerce platform.The graph contains over 215 million nodes and 230 million edges, which is composed of 3 types of nodes and 2 types of edges. It is summarized in Table 2.

Now we provide an overview of this heterogeneous graph.

- **Distribution of nodes' degree**. For each node, its degree suggests how many other nodes linked to it. For example, if a user logins at 3 eids from 6 ips, then its degree is 9. Figure 6 illustrates the distribution of the degrees of each type of nodes.

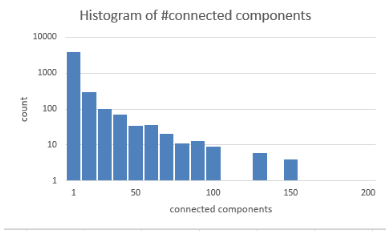
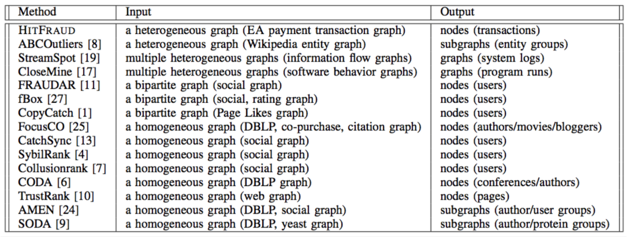


Figure 7: Histogram of size of connected components of heterogeneous graph in login scenario. The majority of connected components is of size $2 \sim 4$.



Method	Input	Output
HitFraud	a heterogeneous graph (EA payment transaction graph)	nodes (transactions)
ABCOutliers [8]	a heterogeneous graph (Wikipedia entity graph)	subgraphs (entity groups)
StreamSpot [19]	multiple heterogeneous graphs (information flow graphs)	graphs (system logs)
CloseMine [17]	multiple heterogeneous graphs (software behavior graphs)	graphs (program runs)
FRAUDAR [11]	a bipartite graph (social graph)	nodes (users)
fBox [27]	a bipartite graph (social, rating graph)	nodes (users)
CopyCatch [1]	a bipartite graph (Page Likes graph)	nodes (users)
FocusCO [25]	a homogeneous graph (DBLP, co-purchase, citation graph)	nodes (authors/movies/bloggers)
CatchSync [13]	a homogeneous graph (social graph)	nodes (users)
SybilRank [4]	a homogeneous graph (social graph)	nodes (users)
Collusionrank [7]	a homogeneous graph (social graph)	nodes (users)
CODA [6]	a homogeneous graph (DBLP graph)	nodes (conferences/authors)
TrustRank [10]	a homogeneous graph (web graph)	nodes (pages)
AMEN [24]	a homogeneous graph (DBLP, social graph)	subgraphs (author/user groups)
SODA [9]	a homogeneous graph (DBLP, yeast graph)	subgraphs (author/protein groups)

Figure 8: Methods on graph based fraud/anomaly/outlier/spam detection listed in [3].

- **Distribution of the size of connected components**. The whole heterogeneous graph is composed of a series of connected components as is illustrated in Figure 7. However, there exists a component of size 190M accounting for over 90% nodes in the graph. And many components are of size $2 \sim 4$, which follows a power law distribution.

4.3 Evaluation

We select several methods [3] that are comparable in our problem domain (See Figure 8). Among these methods, both FRAUDAR [11] and CopyCatch [3] detect suspicious substructures from bipartite graphs. Considering the application scenario, we choose *FRAUDAR* as our baseline method.

We conduct the experiment on the *user-login* graph, and compare the detection accuracy of *HGsuspector* with *FRAUDAR*. At last, we demonstrate the efficiency and scalability of *HGsuspector* on large graphs with billion nodes and edges.

First we extract a smaller graph that contains only users suspected as bulk registry to shrink the large heterogeneous graph as Figure 9. Then decompose it and rate it. finally, we perform a 2-step operation to map a connected bipartite subgraph into a 2-dimensional vector.

A. Decomposition. From dataset 4.2 we find two relation types: *user-eid* and *user-ip* that denote users' devices and locations when they login. In this case, we obtain bipartite graphs: *user-eid* graph and *user-ip* graph as illustrated in Figure 10. The fraud pattern can be described as the fraud users take actions with very specific EIDs and IPs in a short time. And our goal is to detect these users, EIDs or IPs With *HGsuspector*.

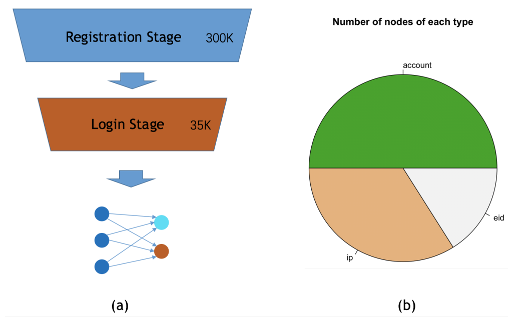


Figure 9: (a) The process of shrinkage. After shrinking, there are only 35K active users in the login stage; (b) The distribution of each type of nodes after shrinkage.

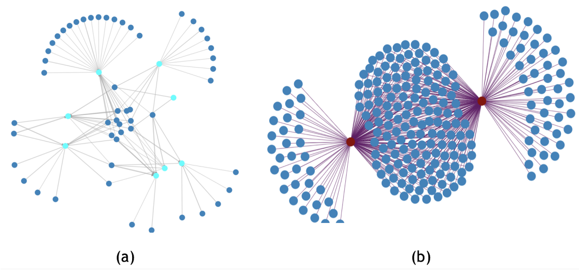


Figure 10: (a): *user-ip* graph. The relations between users and IPs may indicate a fraudulent block; (b) *user-eid* graph. Many users login on two EIDs within 10 minutes, which is very likely to be an attack. *NOTE:* (a) and (b) might share the same nodes, who login on variance of eids and ips.

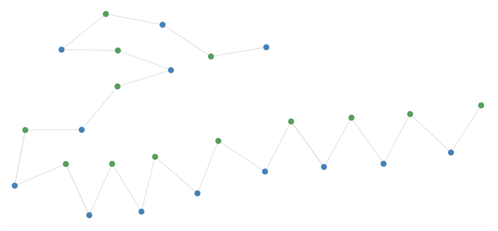


Figure 11: The suspicious pattern found by *HGsuspector*. As shown, fraudsters register many fake accounts (blue) and operate them with fake phones (green) randomly.

B. Scores. With the Equations 3.2 and 3.3, we map each connected bipartite subgraph into a 2-dimensional vector. And employ an anomaly detection algorithm to find suspicious subgraphs.

4.3.1 *Suspicious Patterns.* We run *HGsuspector* on the graphs in login and placing order scenario with edge density function $g(i,j) = \frac{\sum_{i \in S, j \in O} e_{ij}}{m}$. And detect two suspicious patterns which are illustrated in Figure 10 and Figure 11.

4.3.2 *Effectiveness.* Here we show the effectiveness of *HGsuspector* on *user-login* graph, compared with the existing fraud detecting algorithm *FRAUDAR*. The *user-login* graph contains 9,061 validated suspicious users. With these anomalies, we calculate and compare the accuracy of *HGsuspector* and *FRAUDAR*.

For *HGsuspector*, we first score all the connected bipartite subgraphs, and then find the most suspicious subgraphs with *dbscan* [8] algorithm. The statistics of the two bipartite graphs with anomalies is shown in Table 3. Then we compare the precision-recall of *FRAUDAR* and *HGsuspector* on the two bipartite graphs. We first find the subgraphs contain suspicious blocks detected by *FRAUDAR* and the most suspicious subgraphs (the same number of subgraphs detected by *FRAUDAR*) detected by *HGsuspector*. And then compare their precision and recall. The result is illustrated in Figure 12.

For the comparison of *pin-eid* bipartite graph in Figure 12, *HGsuspector* shows low precision when recall< 10%, since the most suspicious subgraphs are much larger than normal ones. The reason is that very few connected components account for over 99% nodes of the *pin-eid* graph. Still, in application scenarios, the large subgraphs can be divided into smaller ones (e.g. communities), which improve the precision.

4.3.3 *Efficiency and Scalability.* Here we show the efficiency and scalability of the methods on a synthetic data set [11], and construct larger graphs by sampling edges from the original one. The induced graphs contain no isolated nodes of degree 0.

We claim that *HGsuspector* can run over large graphs with a upper bound $O(k * M * N)$, which M and N are the numbers of subjects and objects in the largest connected component within a graph, and k is the number of connected components within the graph. We claim that k is decrease, with more and more node types. Thus the upper bound of *HGsuspector* is $O(M * N)$.

The contrast experiments are carried out on an Intel(R) Xeon(R) CPU E5-2640 v3 @ 2.60GHz server, 128 GB RAM, Linux version 2.6.32-431.el6.x86_64, both written in Python. And the result of experiments is provided in Table 4.

Table 3: Statistics of the number of connected bipartite subgraphs with anomalies and all subgraphs.

bipartite graph	#cc	#cc with anomalies	ratio
pin-eid	19,317	4,075	21.1%
pin-ip	22,823	4,838	21.2%

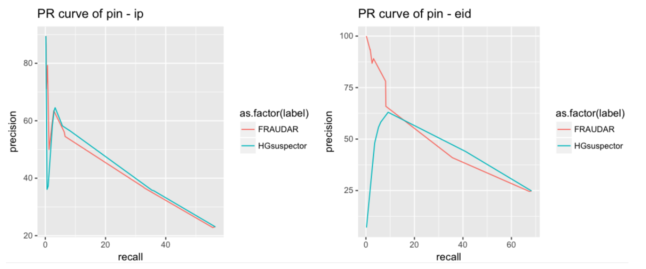


Figure 12: The precision-recall curve of *pin-ip* and *pin-eid* on *FRAUDAR* and *HGsuspector*. It says that with the increasing of blocks detected by *FRAUDAR*, *HGsuspector* partially outperforms *FRAUDAR*.

Table 4: The comparison of running time of *HGsuspector* and *FRAUDAR*, and it is measured in seconds.

#Nodes	*HGsuspector*	*FRAUDAR*
~1K	0.0979	0.4647
~5K	0.3546	2.2188
~10K	0.7053	5.7836
~100K	6.9984	59.0910
~500K	39.5725	303.1607
~1M	81.3569	620.6280
~5M	401.3197	3573.3650
~10M	796.9520	7543.6963

In Table 4, the clustering algorithm *dbscan* in *HGsuspector* is running with parameters $eps = 0.03$ and $MinPts = 8$, and *FRAUDAR* runs with number of blocks 10. The running time of *HGsuspector* contains calculation and scoring of connected bipartite subgraphs. After scores are calculated, we apply *dbscan* or polar coordinate(See Figure 5) to detect the abnormal subgraphs, the time complexity of which is no more than $O(n * log(n))$.

From the result of the comparison, we can see that *HGsuspector* can run over 9 times faster than *FRAUDAR* with the size of graph increasing. In addition, after decomposing of the bipartite graphs, the grading can run in parallel, which dramatically improves the performance of the algorithm. Since *FRAUDAR* builds a priority tree from the degree of a graph, it can not be extended to run in parallel.

Besides the python version of *HGsuspector*, we also developed a distributed version with spark computing framework. In principle, this version of *HGsuspector* could detect suspicious subgraphs of large-scale graphs with billions of nodes and edges, which is impossible for *FRAUDAR*.

5 DISCUSSION AND CONCLUSION

In this paper, we propose *HGsuspector*, a novel algorithm for detecting collective fraud on directed heterogeneous graphs. The algorithm transforms the heterogeneous graphs into bipartite graphs according to our discovery that different types of false information influence the fraud account in respective ways,which also guarantees our algorithm feasible and effective. A metric is proposed to measure the suspicious level of connected bipartite subgraphs, which can distinguish directed graphs with the same structure. In perspective of graph embedding, the metric on connected bipartite graphs can be regarded as a special solution of 2-dimensional embedding. However, our metric is discriminative and thus can be used to detect specific structures compared with methods based on graph embedding.

REFERENCES

[1] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. 2010. OddBall: Spotting Anomalies in Weighted Graphs. In *Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part II (PAKDD'10)*. Springer-Verlag, Berlin, Heidelberg, 410–421. https://doi.org/10.1007/978-3-642-13672-6_40

[2] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph Based Anomaly Detection and Description: A Survey. *Data Min. Knowl. Discov.* 29, 3 (May 2015), 626–688. https://doi.org/10.1007/s10618-014-0365-y

[3] 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 (WWW '13)*. ACM, New York, NY, USA, 119–130. https://doi.org/10.1145/2488388.2488400

[4] Bokai Cao, Mia Mao, Siim Viidu, and Philip S. Yu. 2017. HitFraud: A Broad Learning Approach for Collective Fraud Detection in Heterogeneous Information Networks. *CoRR* abs/1709.04129 (2017).

[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of CIKM*. 891–900.

[6] Ting Chen, Lu An Tang, Yizhou Sun, Zhengzhang Chen, and Kai Zhang. 2016. Entity Embedding-based Anomaly Detection for Heterogeneous Categorical Events. *CoRR* abs/1608.07502 (2016).

[7] Ting Chen, Lu-An Tang, Yizhou Sun, Zhengzhang Chen, and Kai Zhang. 2016. Entity Embedding-based Anomaly Detection for Heterogeneous Categorical Events. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 1396–1403. http://dl.acm.org/citation.cfm?id=3060621.3060815

[8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 226–231. http://dl.acm.org/citation.cfm?id=3001460.3001507

[9] Palash Goyal and Emilio Ferrara. 2017. Graph Embedding Techniques, Applications, and Performance: A Survey. *arXiv preprint arXiv:1705.02801* (2017).

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of KDD*. 855–864.

[11] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 895–904. https://doi.org/10.1145/2939672.2939747

[12] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. CatchSync: Catching Synchronized Behavior in Large Directed Graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 941–950. https://doi.org/10.1145/2623330.2623632

[13] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*.

[14] Shenghua Liu, Bryan Hooi, and Christos Faloutsos. 2017. HoloScope: Topology-and-Spike Aware Fraud Detection. *CoRR* abs/1705.02505 (2017).

[15] M. E. J. Newman. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences* 103, 23 (2006), 8577–8582. https://doi.org/10.1073/pnas.0601602103 arXiv:http://www.pnas.org/content/103/23/8577.full.pdf+html

[16] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric Transitivity Preserving Graph Embedding. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 1105–1114. https://doi.org/10.1145/2939672.2939751

[17] 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 (WWW '07)*. ACM, New York, NY, USA, 201–210. https://doi.org/10.1145/1242572.1242600

[18] Bryan Perozzi and Leman Akoglu. 2016. Scalable anomaly ranking of attributed neighborhoods. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 207–215.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of KDD*. 701–710.

[20] Sujith Ravi and Qiming Diao. 2015. Large Scale Distributed Semi-Supervised Learning Using Streaming Approximation. *CoRR* abs/1512.01752 (2015).

[21] Leonardo F. R. Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. 2017. struc2vec: Learning Node Representations from Structural Identity. In *Proceedings of KDD*.

[22] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip S. Yu. 2015. A Survey of Heterogeneous Information Network Analysis. *CoRR* abs/1511.04854 (2015).

[23] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. *CoRR* abs/1503.03578 (2015).

[24] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of KDD*. 1225–1234.

[25] Jarrod West and Maumita Bhattacharya. 2016. Intelligent Financial Fraud Detection. *Comput. Secur.* 57, C (March 2016), 47–66. https://doi.org/10.1016/j.cose.2015.09.005

[26] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. *CoRR* abs/1603.08861 (2016).