Chapter 14

DETECTING FRAUD IN INTERNET AUCTION SYSTEMS

Yanlin Peng, Linfeng Zhang and Yong Guan

Abstract Fraud compromises the thriving Internet auction market. Studies have shown that fraudsters often manipulate their reputations through sophisticated collusions with accomplices, enabling them to defeat the reputation-based feedback systems that are used to combat fraud. This paper presents an algorithm that can identify colluding fraudsters in real time. Experiments with eBay transaction data show that the algorithm has low false negative and false positive rates. Furthermore, the algorithm can identify fraudsters who are innocent at the time of the data collection, but engage in fraudulent transactions soon after they accumulate good feedback ratings.

Keywords: Internet auctions, fraud detection

1. Introduction

Internet auctions are a major online business. eBay, the leading Internet auction company, had a net revenue of $1.89 billion during the third quarter of 2007; it currently has a community of more than 212 million users around the world. Internet auction fraud can significantly affect this multi-billion-dollar worldwide market. In 2006, the Internet Crime Complaint Center (IC3) reported that Internet auction fraud accounted for 44.9% of all online fraud, and that the average financial loss per case was $602.50 [9].

To prevent potential fraud and encourage honest transactions, Internet auction companies have adopted reputation-based feedback systems. In these systems, users have publicly-viewable feedback ratings, and users may enter comments about each other after completing transactions. A user's feedback score is computed as the number of unique users who have left positive ratings minus the number of unique users

who have left negative ratings. Ideally, fraudsters would have low feedback scores and/or low percentages of positive feedback ratings, discouraging honest users from participating in transactions with them.

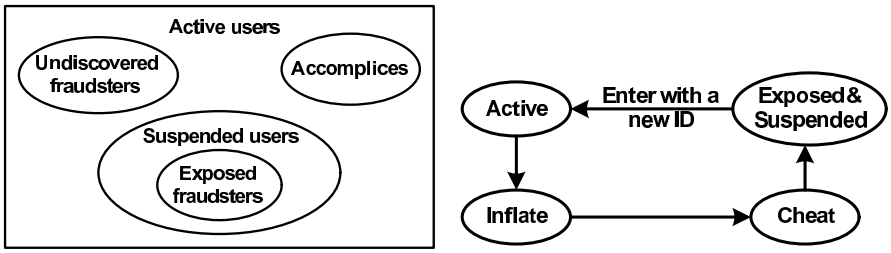
Feedback systems can be manipulated by fraudsters in a variety of ways [3]. The assumption that an honest reputation implies honest behavior in the future is not always valid. Fraudsters often earn good reputations by making several small sales and then make fraudulent transactions on high-priced items. A more sophisticated technique involves collusions with accomplices on "virtual" transactions involving expensive items. After earning a good reputation with 50 or more positive feedback scores, a fraudster can engage in fraudulent transactions with expensive products such as computer equipment. Several researchers (see, e.g., [1, 8]) have attempted to build stronger reputation systems, but these systems are not very effective.

A complementary approach, implemented by eBay's Risk Management Group, is to manually search for fraudulent transactions. However, it is infeasible to investigate every transaction or even a large proportion of the millions of daily transactions on eBay. Some researchers [2, 5] have proposed automated methods that analyze history data to detect abnormal buying and selling behavior. Another strategy [6, 11] is to use belief propagation to identify colluding fraudsters. However, existing approaches are limited or are incapable of detecting sophisticated and hidden relationships between fraudsters and accomplices.

This paper presents an algorithm that identifies – and even predicts – colluding fraudsters in Internet auction systems. When supplied with real eBay data, the algorithm detected twenty fraudsters and predicted ten users as potential fraudsters. The algorithm engages a sliding window to deal with the fact that fraudsters have short lifetimes. The sliding window significantly reduces the computational complexity and makes it possible to accurately process large volumes of transactions in real time. Experiments with synthetic data indicate that algorithm produces very few incorrect identifications.

2. Related Work

Approaches that engage data mining to detect fraud in Internet auction systems fall into two categories: those that detect abnormal patterns of individual users and those that detect sophisticated transaction relationships between users. Bhargava and co-workers [2] have proposed a technique that identifies auction fraud by detecting abnormal profiles and user behavior, building patterns from exposed fraudsters and discovering malicious intentions. Chau and colleagues [5] have developed a



(a) Internet auction community. (b) State transitions of fraudsters.

Figure 1. Internet auction community and state transitions of fraudsters.

data mining method that generates a decision tree based on sixteen features extracted from user profiles and individual transaction histories; the decision tree is used to classify users as "legitimate" or "fraudulent." A more sophisticated method for detecting abnormal transaction relationships between users [6] uses belief propagation to discover abnormal transaction patterns between colluding fraudsters and accomplices (modeled as bipartite subgraphs in a undirected transaction graph). An improved technique [11] uses incremental belief propagation on a smaller subgraph (three-hop neighborhood) for each new transaction. However, all these methods are unable to detect collusion in Internet auction systems efficiently and accurately.

The problem addressed in this paper is similar to the dense subgraph detection problem in web graphs. Two algorithms for detecting these graphs are "trawling" [10], which enumerates all the complete bipartite subgraphs; and "shingling" [7], which extracts dense subgraphs. However, these algorithms do not address the problem presented in this paper in an efficient manner.

3. Fraud Model and Problem Definition

This section discusses the fraud model underlying Internet auction systems and defines the fundamental problem addressed in this paper.

3.1 Fraud Model

Figure 1(a) shows a classification of user accounts (also called "users") in an Internet auction community. Every person who registers successfully receives an "active" user account. Some registered users may be suspended and cannot conduct transactions for various reasons (e.g., for committing fraudulent transactions). A "Not-A-Registered-User" message is shown on the profile page of suspended users. Exposed fraudsters are placed in the category of suspended users. Some users are

"accomplices," who do not conduct fraudulent transactions directly, but may help fraudsters inflate their positive ratings. Additionally, there are "undiscovered fraudsters" who have not committed fraud as yet and are, therefore, active users.

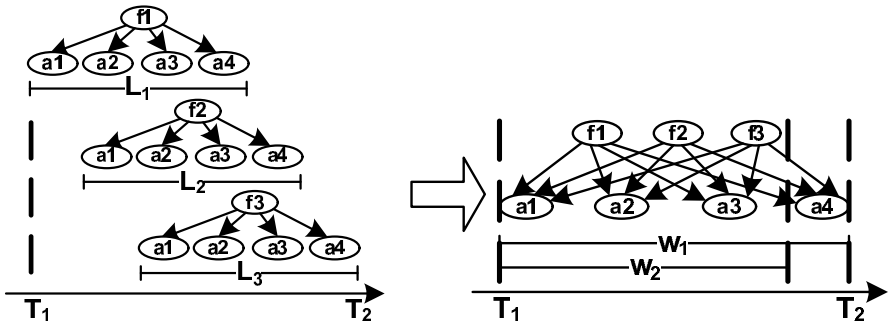
A user may engage in selling fraud or buying fraud. This paper focuses on selling fraud. However, the proposed scheme can be applied to buying fraud with slight modifications.

A fraudster may create multiple user accounts of two types, fraudster accounts and accomplice accounts. Fraudster accounts are used to commit fraud after good reputations have been earned. A fraudster's account may be suspended after a fraudulent transaction, but by then the fraudster may have already made a profit and could return as a new user. Figure 1(b) shows the state transitions made during the lifetime of an auction fraudster. Note that the lifetime of a user account ranges from the time it is created and registered to the time it is suspended. The accomplice accounts, which cannot be identified by current detection methods, remain as legitimate, active users. Accomplices typically serve multiple fraudsters. Therefore, the relationships between fraudsters and accomplices can be expressed as "bipartite cores" (i.e., complete bipartite subgraphs) in a transaction graph.

3.2 Problem Definition

We model the relationships between users in an Internet auction system as a directed transaction graph and colluding patterns as bipartite cores. A bipartite core is a complete bipartite subgraph consisting of two groups of nodes A and B. Every node in A is connected to all the nodes in B. Detecting colluding fraudsters from transaction data is equivalent to detecting bipartite cores with size $\geq s \times t$ in the transaction graph. A transaction graph is a directed graph $G = (V, E)$, where V is the set of nodes representing users and E is the set of directed edges representing transactions between two users. A bipartite core consists of two sets of nodes, "parent nodes" (sellers) and "child nodes" (buyers). Each edge is a transaction with a timestamp T that denotes when the transaction occurred. Edges are added to the transaction graph in chronological order.

One method for extracting bipartite cores is to process the entire transaction graph. However, based on our analysis of eBay data, fraudsters usually have short lifetimes because they do not wait long before committing fraud. Once they commit fraud, they are quickly reported and suspended. Consequently, detecting fraudsters only requires the extraction of the bipartite cores from a subgraph of the transaction graph.



(a) Fraudster-accomplice transactions. (b) Bipartite cores.

Figure 2. Sliding window model and bipartite cores within windows.

In the example in Figure 2, we assume that a fraudster f_i $(i = 1, 2, 3)$ has a lifetime L_i. During their lifetimes, fraudsters engage in transactions with accomplices a_j $(j = 1, 2, 3, 4)$. The earliest transaction between them occurs at time T_1 and the latest at time T_2 $(T_1 < T_2)$. If the bipartite core is extracted from a subgraph of transactions bound by a window (or time interval) $w_1 \geq (T_2 - T_1)$, the collusion forms a 3×4 bipartite core (Figure 2(b)). If a smaller window of size w_2 is chosen, the subgraph is smaller and only some of the fraudsters' transactions are included; the resulting bipartite core is 3×3. Based on our analysis of eBay data, a three-month window size is sufficient to identify most fraudsters.

However, not all bipartite cores indicate collusion. Sometimes, small bipartite cores are normal patterns among honest users, especially when the bipartite cores exist in the neighborhood of power users who conduct large numbers of transactions with their customers. Consequently, our algorithm excludes power users and small bipartite cores in order to capture fraud patterns more accurately.

In summary, this paper focuses on the problem of detecting all bipartite cores with size $\geq s \times t$ that represent collusion relationships between fraudsters and accomplices from a large dynamically-changing transaction graph G.

4. Fraud Detection Algorithm

In the preliminary step of the algorithm, non-positive feedback ratings are removed because fraudsters and accomplices always leave positive ratings for each other. The detection process then involves three steps. In the first step, transactions that are outside the sliding window with respect to the newly arrived transaction are removed. The second step

performs filtering, including the removal of power users and common neighbors. The common neighbor filter retains users who share the trait of having purchased from the same (two or more) sellers. The third step of the algorithm computes and reports the bipartite cores for detecting fraudsters.

Three data structures are used in the algorithm:

- **Transaction Storage:** This includes a FIFO queue Q_t and a hash table H_t. Q_t stores transaction entries with timestamps in ascending order. H_t stores the pointers of transaction entries in Q_t using the trader IDs as keys.

- **Common Neighbor Counter Storage:** This hash table H_c stores the number of common neighbors. The common neighbors for a pair of parent nodes (i, k) is the intersection of their child nodes $N(i) \cap N(k)$, where $N(\cdot)$ is the set of neighboring child nodes. H_c stores $\langle i, k, C_{i,k} \rangle$ using (i, k) as keys (where $C_{i,k} = |N(i) \cap N(k)|$).

- **Bipartite Core Storage:** This hash table H_b stores the detected maximum bipartite cores using the IDs of nodes in the bipartite core as keys. A maximum bipartite core is a core that is not a subset of another bipartite core. Let $G_p(b)$ denote the group of parent nodes in a bipartite core b and $G_c(b)$ denote the group of child nodes. For any two bipartite cores b_1 and b_2, if $G_p(b_1) \subseteq G_p(b_2)$ and $G_c(b_1) \subseteq G_c(b_2)$, only b_2 is stored; otherwise, both bipartite cores are stored.

4.1 Filtering

Two filtering functions are used to efficiently discard edges that do not contribute to qualified bipartite cores. The Power-User-Filter(i, j, R) function removes users whose reputations exceed R. Fraudsters rarely spend the time to earn extremely high reputations as lower reputations suffice for their purposes. The Common-Neighbor-Filter(i, j, t) function checks for at least t common neighbors (buyers) for a pair of parents (sellers) prior to building $s \times t$ bipartite cores. For each transaction (i, j) and parent pair (i, k) (where j is also a child of k), the child j is added to the set $N(k)$ and the common neighbor counter for k is incremented. If the maximum common neighbor counter for a parent node is at least t, the filter returns "pass" and the process enters the next step to compute the bipartite cores.

Algorithm 1 Compute-Bipartite-Cores

input: new transaction (i, j), detection size t

1: **for all** k such that parent pair $(i, k) \in H_c$ **do**
2: **if** $C_{i,k}$ is increased to t after adding (i, j) **then**
3: new bipartite core $b \leftarrow \{\{i, k\}, N(i) \cap N(k)\}$
4: **if** $b \notin H_b$ **then**
5: $Insert(H_b, node, b)$
6: **end if**
7: **end if**
8: **end for**
9: **for all** bipartite core $\widehat{b} \in H_b$ **do**
10: **if** $|G_p(\widehat{b}) \cap N(j)| \geq 2$ **then**
11: new bipartite core $b \leftarrow \{G_p(\widehat{b}) \cap N(j), G_c(\widehat{b}) \cup \{j\}\}$
12: **else if** $|G_c(\widehat{b}) \cap N(i)| \geq t$ **then**
13: new bipartite core $b \leftarrow \{G_p(\widehat{b}) \cup \{i\}, G_c(\widehat{b}) \cap N(i)\}$
14: **end if**
15: **if** $b \notin H_b$ **then**
16: $Insert(H_b, node, b)$
17: **end if**
18: **end for**

4.2 Computing Bipartite Cores

The process of computing bipartite cores builds larger bipartite cores from smaller ones. As shown in Algorithm 1, the smallest $2 \times t$ bipartite cores are constructed by intersecting the parent pairs whose common neighbor counters are no less than t (Lines 1–8). Then, the bipartite cores containing i or j are retrieved to check if they can be enlarged. For example, if a bipartite core b contains node i, then the intersection of $G_p(b)$ and $N(j)$ is checked. If the size of the intersection is large enough, the existing bipartite core b can be enlarged by adding the child node j. If an enlarged bipartite core is not a subset of an existing core, it is added to the bipartite core storage.

4.3 Reporting

The reporting function lists the users present in "fraudulent bipartite cores." A fraudulent bipartite core is a bipartite core with size $\geq s \times t$ that contains at least one exposed fraudster. We believe that users in these bipartite cores will commit fraud with a high probability. Non-

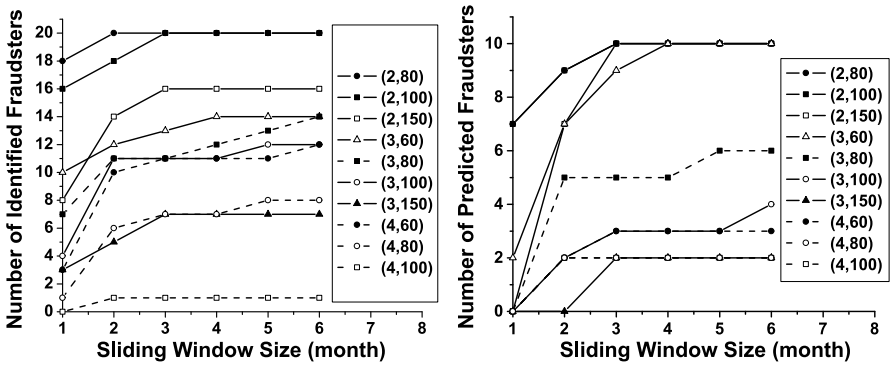
fraudulent bipartite cores are also stored. As soon as a user is identified as a fraudster, all the other users in the bipartite core are reported.

5. Analysis of Countermeasures

Armed with the details of the algorithm, fraudsters can implement two countermeasures to evade detection. In the following discussion, we assume that: (i) n accomplices support as many fraudsters as possible and evade detection; (ii) a fraudster needs to earn a reputation of at least r before committing fraud; and (iii) the detection size is (s, t) and $t \leq r \leq n$.

The first countermeasure is to form bipartite cores of size $i \times j$ where $i \geq s, j < t$. Thus, less than t accomplices support at least s fraudsters, which is not detected. However, a fraudster cannot earn a reputation of r from one group of j accomplices within a sliding window. The fraudster must wait longer than the time period covered by one sliding window to earn the required reputation or he must collude with additional accomplices. Even if the fraudster waits longer to earn a reputation of r, the system administrator can enlarge the sliding window to counter this strategy. If the fraudster attempts to earn the desired reputation within one sliding window, a subset of the i fraudsters must collude with other accomplices, forming bipartite cores with size $f \times g$ where $f < s, g \geq r$. This is the same as the second countermeasure.

The second countermeasure is to form bipartite cores of size $i \times j$ where $i < s, j \geq r$. Thus, at least r accomplices support fewer than s fraudsters. Such collusion is not detected and a fraudster can earn a reputation of r within a short time. However, the proposed algorithm can still set an upper bound on the number of supported fraudsters in this case. To maximize the number of supported fraudsters, it is necessary to maximize the number of j-accomplice groups, maximize the number of fraudsters supported by each accomplice group, and support different fraudsters for each accomplice group. Assume that there are m ways to choose accomplice groups such that each group has at least r accomplices and every two groups have intersections of at most $(t-1)$. According to the non-uniform Ray-Chaudhuri-Wilson inequality, $m \leq \binom{n}{t-1} + \cdots + \binom{n}{0}$. Thus, at most $m \cdot (s-1)$ fraudsters can be supported by a group of n accomplices without being detected. To address this countermeasure, the systems administrator can choose small values of s and t to reduce the number of fraudsters that can be supported by an accomplice group.



(a) Number of reported fraudsters. (b) Number of predicted fraudsters.

Figure 3. Results obtained with eBay data.

6. Experimental Results

This section describes the experimental results obtained using real eBay data and synthetic data.

6.1 Results with eBay Data

The first set of experiments used eBay data collected from September 3, 2007 to November 3, 2007. The data was gathered by a crawling program that started with a list of eBay user IDs and obtained their feedback profiles in a breadth-first manner. In all, 5,795,314 transactions and 3,406,783 eBay users were crawled. Of these users, 43 were fraudsters who received many negative ratings and were suspended. Since the identified fraudsters may not have colluded with accomplices, our algorithm may only identify some of the fraudsters.

The algorithm was executed on a 2 GB dual core Mac Pro. Transactions were input in ascending order of transaction timestamps. The power user threshold was set to 3000. Several detection sizes and sliding window sizes were tested in our experiments.

Figure 3(a) shows that as many as twenty of the identified fraudsters were reported by the algorithm. Figure 3(b) shows the most important result – up to 10 users were predicted to be fraudsters. These users were not identified as fraudsters at the time the data was collected. After the users were reported by the algorithm, we went back to check them out and discovered that they had received many negative ratings and had been suspended. If our algorithm had been applied in a real-world setting, these users would have been flagged before they committed fraud.

Note also that the number of reported fraudsters and the number of predicted fraudsters increase when the window size increases. The reason is that a larger sliding window produces larger subgraphs, which results in more fraudulent bipartite cores being detected. Interestingly, regardless of bipartite core size, the number of detected fraudsters is maximum when a three-month sliding window is used (Figure 3(a)). This confirms our assumption that fraudsters have short lifetimes.

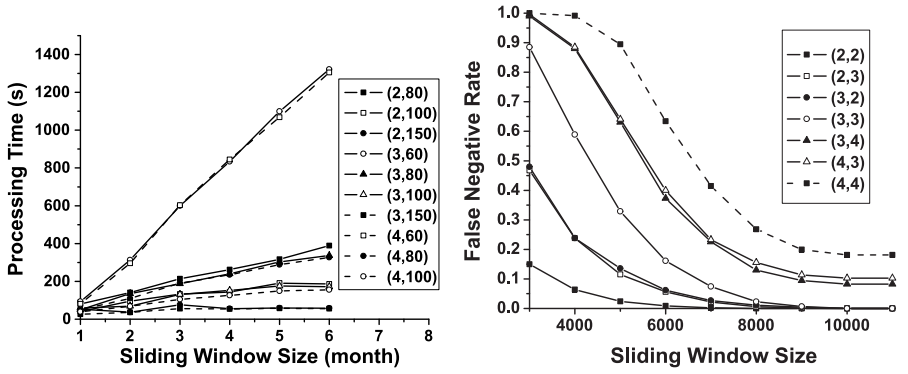
Figure 4(a) shows the increase in processing time with respect to window size. From these results, it is evident that the best setting for the sliding window is three months because it provides good detection accuracy with lower processing overhead.

Figure 3 shows that the number of detected fraudsters also depends on the bipartite core detection size. For larger sizes, fewer fraudsters are identified because fraudsters in small collusions are not detected. The detection sizes (2, 80) and (2, 100) identify the most fraudsters. Also, the processing time increases for larger detection sizes. Based on these results, we recommend that 2×100 detection sizes be used for fraud detection.

6.2 Results with Synthetic Data

Synthetic data was used to evaluate the false positive and false negative rates for the algorithm. The false negative rate is defined as the number of distinct nodes in the injected bipartite cores that are not reported divided by the number of total distinct nodes in all the injected bipartite cores. The false positive rate is defined as the number of distinct nodes not in the injected bipartite cores but that are reported divided by the number of total distinct nodes not in any injected bipartite core.

Since the distribution of eBay feedback obeys a power law distribution [12], R-MAT [4] was used to generate a random power law transaction graph of about 100,000 nodes with an average degree of 1.7. Next, ten bipartite cores of known sizes were injected as fraudster bipartite cores as follows: (i) the sizes of the two groups were randomly chosen to be between 3 and 10; (iii) nodes in each group were randomly chosen among all nodes; and (iii) a fraud lifetime of 10,000 units was defined to represent a short lifetime. The transaction timestamps in each fraudulent bipartite core were chosen randomly within the fraudster lifetime. Some nodes in these bipartite cores were randomly selected as identified fraudsters. Then, the transaction graph was input to the algorithm and the fraudsters were reported.



(a) Processing time. (b) False negative rates.

Figure 4. Processing time (eBay data) and false negative rates (synthetic data).

Figure 4(b) shows the average results over 500 executions. Note that the false negative rate decreases when the window size increases. A larger window ensures larger subgraphs and more fraudulent cores; this reduces the likelihood of missing fraudulent nodes. When the window size is equal to the fraud lifetime, the false negative rate reaches its lowest value for each detection size. Note also that the false negative rate decreases when the detection size decreases. Most of the fraudulent nodes are detected even when a very small window size of 0.4 times the fraud lifetime and a detection size of 2×2 are used.

The false positive rates are close to zero for all window sizes and detection sizes. Honest nodes are wrongly reported only if they happen to be in a fraudulent core, which has a very small probability. Thus, our algorithm can distinguish honest users from fraudsters very effectively. Moreover, the sliding window method reduces not only the processing overhead but also the false positive rate.

7. Conclusions

Fraudulent activities can compromise the multi-billion-dollar Internet auction market. Existing solutions are either limited or are incapable of detecting colluding fraudsters. The algorithm presented in this paper can detect colluding fraudsters at runtime with good accuracy. Experiments with real and synthetic data demonstrate that the algorithm can detect fraudsters and, more importantly, predict potential fraudsters.

References

[1] S. Ba, A. Whinston and H. Zhang, Building trust in online auction markets through an economic incentive mechanism, *Decision Support Systems*, vol. 35(3), pp. 273–286, 2003.

[2] B. Bhargava, Y. Zhong and Y. Lu, Fraud formalization and detection, *Proceedings of the Fifth International Conference on Data Warehousing and Knowledge Discovery*, pp. 330–339, 2003.

[3] M. Calkins, My reputation always had more fun than me: The failure of eBay's feedback model to effectively prevent online auction fraud, *Richmond Journal of Law and Technology*, vol. VII(4), 2001.

[4] D. Chakrabarti, Y. Zhan and C. Faloutsos, R-MAT: A recursive model for graph mining, *Proceedings of the SIAM International Conference on Data Mining*, 2004.

[5] D. Chau and C. Faloutsos, Fraud detection in electronic auctions, *Proceedings of the European Web Mining Forum*, 2005.

[6] D. Chau, S. Pandit and C. Faloutsos, Detecting fraudulent personalities in networks of online auctioneers, *Proceedings of the Tenth European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 103–114, 2006.

[7] D. Gibson, R. Kumar and A. Tomkins, Discovering large dense subgraphs in massive graphs, *Proceedings of the Thirty-First International Conference on Very Large Data Bases*, pp. 721–732, 2005.

[8] D. Houser and J. Wodders, Reputation in auctions: Theory and evidence from eBay, *Journal of Economics and Management Strategy*, vol. 15(2), pp. 353–369, 2006.

[9] Internet Crime Complaint Center, Internet Crime Report 2006 (www.ic3.gov/media/annualreport/2006_IC3Report.pdf), 2006.

[10] R. Kumar, P. Raghavan, S. Rajagopalan and A. Tomkins, Trawling the web for cyber communities, *Computer Networks*, vol. 31(11-16), pp. 1481–1493, 1999.

[11] S. Pandit, D. Chau, S. Wang and C. Faloutsos, Netprobe: A fast and scalable system for fraud detection in online auction networks, *Proceedings of the Sixteenth International Conference on the World Wide Web*, pp. 201–210, 2007.

[12] M. Zhou and F. Hwang, PowerTrust: A robust and scalable reputation system for trusted peer-to-peer computing, *IEEE Transactions on Parallel and Distributed Systems*, vol. 18(4), pp. 460–473, 2007.