A DYNAMIC STAGE-BASED FRAUD MONITORING FRAMEWORK FOR MULTIPLE LIVE AUCTIONS

A Thesis

Submitted to the Faculty of Graduate Studies and Research

In Partial Fulfilment of the Requirements

For the Degree of

Masters of Science

In

Computer Science

University of Regina

By

Xuegang Wang

Regina, Saskatchewan

April, 2016

UNIVERSITY OF REGINA

FACULTY OF GRADUATE STUDIES AND RESEARCH

SUPERVISORY AND EXAMINING COMMITTEE

Xuegang Wang, candidate for the degree of Master of Applied Science in Computer Science**,** has presented a thesis titled, ***A Dynamic Stage-Based Fraud Monitoring Framework for Multiple Live Auctions,*** in an oral examination held on April 21, 2016. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner:	Dr. Mohamed Ismail, Faculty of Engineering and Applied Science
Co-Supervisor:	Dr.Malek Mouhoub, Department of Computer Science
Co-Supervisor:	Dr. Samira Sadaoui, Department of Computer Science
Committee Member:	Dr. Xue Dong Yang, Department of Computer Science
Chair of Defense:	Dr. Maria Velez, Department of Geology

ABSTRACT

In this thesis, a new approach to monitoring live auction frauds is proposed. Monitoring progressing auctions for fraudulent bidding activities is becoming crucial in order to detect and stop fraud on time, so fraudsters will not succeed. For this purpose, we introduce a generic stage-based framework to monitor multiple live auctions for In-Auction Fraud. Creating a stage-based runtime fraud monitoring service is substantially different than the very limited studies that have been proposed on runtime fraud detection.

More precisely, we launch the fraud monitoring operation at several time points in each running auction depending on its duration. At each auction time point, our framework first detects fraud by evaluating each bidder's behaviour by using the most reliable set of fraud patterns together with our stage-based fraud detection method. The framework then reacts to malicious activities by taking proper actions. We develop the proposed framework with a dynamic architecture where multiple monitoring agents can be created or deleted with respect to the status of their corresponding auctions (initialized, completed or cancelled). Adopting a dynamic software architecture represents an excellent solution to handle the scalability and real-time performance issues of fraud monitoring systems since hundreds of auctions are performed simultaneously in commercial websites. Every time an auction ends, successfully or not, the participants' fraud scores and clusters are updated.

We validate our fraud monitoring service through commercial auction data. We

conduct three experiments to detect and react to shill bidding fraud by employing auction datasets of two valuable items, Palm PDA and XBOX. Actually, we observe each auction at three time points, and for each of them we verify the collection of shill patterns that most likely happen in the corresponding stage.

ACKNOWLEDGEMENT

I would like to address my sincere gratitude to my supervisors Dr. Samira Sadaoui and Dr. Malek Mouhoub for the financial assistance, precious comments and remarks, and constant encouragement through my M.Sc. studies. I would like to acknowledge Dr. Samira Sadaoui once more for her continuous help on my thesis in the past. Without her encouragement and effort, this work would not have been completed.

Beside my supervisors, I would like to thank my committee member Dr. Xuedong Yang for taking time to read my thesis and providing valuable suggestions and useful comments for improvement.

I also thank the Faculty of Graduate Studies and Research for providing me scholarships to complete my studies.

I thank every member in the Department of Computer Science who ever has helped me in the past.

Last but not the least, I would like to extend my deepest thanks to my family: my parents, elder brother and younger sister for supporting me financially and spiritually throughout my whole studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Algorithms

Chapter 1

Introduction

1.1 Scope and Motivation

In the context of online auctions, several topics have been identified in the mainstream research: eliciting preferences for multi-criteria auctions, designing various auction mechanisms, determining the winners for different auction protocols, and monitoring auctions for fraud. In the commercial auctions (mostly forward and price-only attribute), a voluminous number of users are bidding daily. For instance, in the year 2014, eBay recorded a total of 115 million users, and held around 265 million auctions [1]. In spite of many advantages of online auctioning, serious threats menace the users' interests. Due to the huge traffic, online auctions attract a large number of fraudsters [2, 3]. As reported in [4], auction fraud remained in the top two of cybercrimes since 2004. Moreover, the federal Internet Crime Complaint Center (IC3) estimated a money loss of $11 million from June to December 2014 [5] due to auction fraud. According to the IC3 report of 2013, 5% of the Internet complaints are related to the automobile auction scams with a loss of $51 million in 2013, an increase of $43 million when compared to 2011.

Online auctions give many opportunities for conducting misbehaviour. Some risks may happen before and after the auctions, such as misrepresented and non-existent items, non-payment and non-delivery of items. Additionally, fraudulent activities may occur during the bidding period, called In-Auction Fraud (IAF) [2]. IAF, such as shill bidding, is hard to detect unlike pre- and post-auction crimes. A serious concern is that the innocent bidders are not even aware about the committed IAF [6]. Both the auctioneers and bidders may conduct fraud. Dishonest users utilize various IAF strategies that have been recognized in the English forward auctions [6, 7], including 1) shilling, i.e., inflating the price by placing false bids in order to generate an interest in the item and persuade other participants to bid more; 2) shielding, i.e., placing a very high bid and then withdrawing it from the auction prior to the closing time in order to purchase the item at a low price; 3) sniping, i.e., submitting a bid in the final seconds of the auction to guarantee oneself to win; 4) user collusion such as between a seller and a bidder, or between bidders; 5) siphoning, i.e., a seller who does not want to pay the auction fees watches an ongoing auction in order to propose a lower price to the winning buyers. Some IAF behaviour may be prevented by implementing rules within the negotiation protocol, such as shielding by disallowing bidders from withdrawing bids, and sniping by extending the auction duration. Nevertheless, other IAF types, such as shill bidding and user collusion, must be monitored as they cannot be prevented in advance. Shill bidding has been recognized as one of the most dominant cheating activities in online auctions, and also the hardest one to detect [8, 9]. To push the winner to pay more for the auctioned item, a seller may manipulate his own auction, for example by bidding via alternate identities (with fake accounts and IP addresses) [7]. In a fraud-infected auction, the difference between the final and normal auction prices is the seller's revenue [6]. Several empirical studies have demonstrated the presence of shill bidding by examining offline auction data from popular auction houses. As an example, [8, 9] analyzed bidding data of eBay and revealed that several shilling strategies have been frequently used.

Consequently, monitoring online auctions for IAF is becoming crucial. We categorize existing auction monitoring solutions into two types: offline and runtime. Offline or batch monitoring models extract, filter and then analyze a numerous amount of historical transactions of auction [8, 10]. Although batch analysis may find numerous IAF patterns, it is very time consuming. Besides, it is too late because offline detection is performed after the crime has occurred and the auctions have already resulted in money loss and wasted time for honest buyers [11]. Since the damage happens during the auction, it is critical to detect and stop IAF in runtime so fraudsters will not succeed. So far, all existing auction houses in the e-market, and almost all the precedent research works did not implement services that detect IAF in progressing auctions, and did not take runtime auctions against malicious bidders.

1.2 Contributions

To increase trust in e-auctions, it is indispensable to detect undesirable bidding activities on time before it is too late. We introduce a generic stage-based framework that covers the runtime monitoring of IAF for multiple progressing auctions as depicted in Figure 1.1. Developing a stage-based runtime fraud monitoring service is substantially different than has been proposed in the very limited studies on runtime fraud detection. In those studies, the fraud detection is performed after every single submitted bid, and every bid is evaluated against the same set of fraud patterns. This will definitely make the fraud detection very time-consuming due to unnecessary pattern evaluation. In our work, we believe that a bidder cannot be blamed on a single bid but on a series of bids. The more we include evidence of user's misbehaviour, the more we increase the fraud detection accuracy.

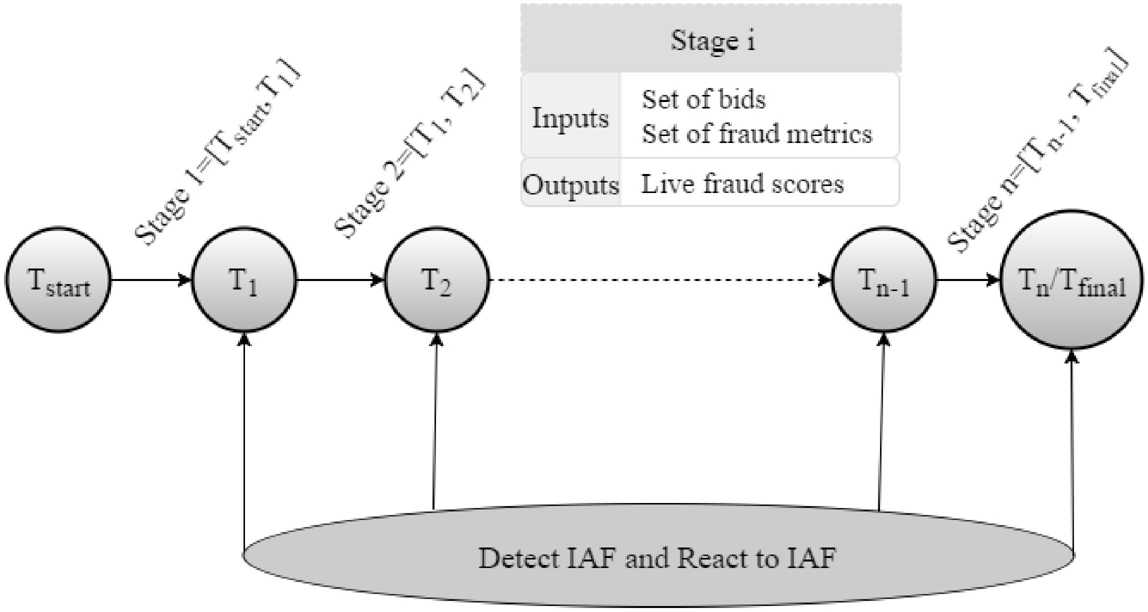

Figure 1.1: Stage-based Fraud Monitoring of a Live Auction.

More precisely, we launch the monitoring operation at several time points of the running auctions. We divide the monitoring functionality into two tasks: detecting IAF and reacting to IAF. The first task examines bidding activities at each auction time point based on a set of fraud patterns, and the second one determines how to react to illegal activities by taking runtime decisions against the fraudsters and infected auctions as well. We quantify the IAF patterns with metrics and aggregate them with a certain fusing technique to produce an overall value representing the live fraud score of each bidder in each ongoing auction. We observe a bidder in each auction as he may change his behaviour in just few specific auctions. The live fraud score is updated at each auction time point by including the set of fraud metrics of the corresponding stage. To improve the detection accuracy, we consider properties from different sources: users, auctions and bids. Each time an auction is completed successfully or terminated, we combine the live fraud score with the past fraud score to generate a comprehensive value denoting the current level of misconduct of a bidder in the auction community. Likewise, we associate auction users with clusters (normal, suspicious or fraudulent) corresponding to their current fraud scores. In this way, we will have a very good idea about each user's behaviour in the auction community. These clusters, shown to all to see, may be utilized as a deterrent for bidders to commit IAF. Reputation models in e-commerce applications are based on feedback ratings that can be easily manipulated. In our point of view, the user's fraud score may be used as a strong evidence towards his reputation.

The main work of my thesis are as follows: 1) based on the agent technology, we design our auction monitoring framework as a dynamic multi-agent system where both detection and reaction agents are added and removed during runtime. The creation and destruction of these agents correspond respectively to the auction status: initialized, completed or terminated. Adaptive software architecture represents an excellent solution to handle the scalability and time performance issues of fraud monitoring systems since hundreds of auctions are performed simultaneously in commercial auctions; 2) we implement fully our monitoring system by employing an agent simulation platform where asynchronous communication and dynamic creation and deletion of agents are well supported. Our system can monitor a very large number of auctions. Moreover, this implementation is specifically based on the shill bidding fraud; 3) we validate the proposed monitoring service through commercial auction data. We conduct three experiments to detect and react to shill bidding fraud by employing auction datasets of two valuable items, Palm PDA and XBOX. Actually, we observe each ongoing auction at three time points, and for each of them we verify the collection of shill patterns that most likely happen in the corresponding stage.

We may note that in our IAF monitoring system, the time performance is not an issue for three reasons:

1. Each auction is assigned an agent to monitor it.

2. A smaller set of stage bids (stage-based bids) is processed as opposed to offline fraud detection where an enormous volume of auction data is evaluated at once.

3. Commercial auctions last several days and our monitoring algorithm will take only few seconds to respond while the bidders are still competing at the very beginning of the current stage.

1.3 Thesis Organization

This thesis is organized as follows. Chapter 2 covers the relevant works of our research, including the shill bidding strategies and categorization as well as their stages in the live auctions. This chapter also provides the exiting solutions on shill detection approaches, which are divided into offline and runtime detection. Finally it presents the auction fraud detection systems that employs commercial auction data for their evaluation.

In Chapter 3, we describe in detail our dynamic and generic auction fraud monitoring system. Firstly, we expose the software architecture and workflow of the whole auction system. Secondly, we present the dynamic features and agent location mechanisms of our auction agents. Lastly, we propose several algorithms: live auction monitoring, in-auction fraud detection, reaction to in-auction fraud, and cluster updating of auction users.

In Chapter 4, the implementation with the shill bidding fraud is discussed. It first introduces 17 shill bidding patterns which are divided into three stages. Among those 17 shill bidding patterns, eight metrics are proposed as well as their combination technique. At last, it describes the auction system implementation based on the BDI model and a simulation platform of the agent community.

To validate the proposed auction fraud monitoring service, three experiments are conducted in Chapter 5. It presents statistical features of real commercial online auction datasets. Three live auctions are monitored. Finally, the results of the experiments are discussed.

Finally, Chapter 6 comprises two sections. The conclusions and future works of this research work.

Chapter 2

Related Works

Most of the studies on In-Auction Fraud (IAF) detection focused on the shill bidding. This chapter is divided into three sections. Section 2.1 explains the shill bidding strategies and stages in a certain auction. In Section 2.2, the shill bidding detection mechanisms, which are categorized into two types: offline and runtime detection. Finally, in Section 2.3, we report the shill detection systems that utilized commercial auction data.

2.1 Shill Bidding Strategies

The goal of shill bidding is to increase the price of goods or services in case of forward auctions in order to generate the interest for the auctioned items but without winning the auctions. For instance, the seller would get an associate user to compete in his own auction to make the item looks more popular than it actually is, or would create an alternate account to commit the fraud [7]. Diverse shill strategies have been identified and recognized in the English forward auction protocol, for example six stragegies in [8], three in [13], five in [14], nine in [11], one in [15], three in [16], and eight in [10]. Some of these shill patterns are similar across these papers because they appear more often in the auction data. We classify all the shill bidding patterns into several groups. In Table 2.1, a Shill Bidder (SB) represents a seller or an accomplice bidder.

Strategy	**Description**
Security related	An SB creates fake identities by using different accounts and IP addresses.
Collusive behaviour	An SB participates exclusively in auctions held by some particular sellers, colluding SBs work together to inflate the price in an auction, SBs place bids on each other's auctions, or SBs who live in a proximity area collude.
Competitive shilling	An SB aggressively increases the price, or bids more often.
Reputation manipulation	SBs collude by helping each other building a good reputation by submitting positive ratings.
Buy-back shilling	An SB wins the auction to re-sell the item in case the current auction price is low.

Table 2.1: Shill Bidding Strategies.

A shill bidding pattern may mostly occur in a certain auction stage. [11] and [2] proposed three stages:

- Early stage, i.e., in the first 25% of the elapsed auction time because a SB places bids very early in the auction to encourage others to bid, especially when the participation rate is low.
- Middle stage, i.e., from 25% to 90% of the auction duration since most of the bidding activities happen at this stage.
- Last stage, i.e., in the last 10% of the auction time because a SB submits very few bids. Bidding towards the end of an auction is very risky as the fraudster could accidentally win.

2.2 Shill Bidding Detection

Some prevention approaches have been adopted to try to deter users from committing auction fraud, such as requiring credit card information when registering to the auction site [17] [18], or taking into account the users' reputation in the auction system [11]. Even so, they are not enough against shill bidding. We split the auction fraud detection research into twofold: offline V.S. runtime detection. Almost all the methods on shill detection are done offline. Indeed, very limited studies proposed shill detection services in runtime shill detection services.

2.2.1 Offline Detection

In the literature, numerous offline detection models have been proposed. There are two major drawbacks of offline approaches: the analysis of a tremendous amount of batch auction data, which is time-consuming; it is too late to react to shills as the innocent bidders have been already cheated.

Several approaches are based on mathematical theories.

[7], [8], and [19] defined the measurements for a set of shill patterns and aggregated them to produce a final shill score. To demonstrate shill bidding impacts on real auction price and expected auction price, [19] analyses the shill bidders' behaviours using shill scores to verify hypotheses set. Authors want the auction bidders could deduce shill behaviours exclusively through the final price. Trevathan and Read explore the shill characteristics and strategies thoroughly, six of which are made use of calculating shill scores. [8] conducts experiments based on 39 simulated auctions and 150 real auctions from eBay. Nevertheless, their algorithm copes with only one shill bidder in one auction.

[11] formalized and then verified patterns of shill bidding by using a formal specification method and model checking. [10] employed Dempster-Shafer theory to

express shill patterns as pieces of evidence and combine them to provide the degree of belief of a bidder for shilling. The evidences for supporting a shill are partitioned into two levels: bid-level and auction-level. The bid-level evidences connect with a particular bidder, while auction-level evidences associate with all the participants in the auction. The outcome of combination evidences reflects the shill activities in the auction and will be used to categorize bidders into several groups.

Moreover, several supervised learning techniques have been used for shill bidding detection.

[19] proposed an approach based on the artificial neural networks in order to predict the final auction price according to the shill activities. Other papers utilized decision trees to classify bidders into two groups: "Regular" and "Shill Bidder" [3, 17, 20, 21], or into three groups: "Normal", "Suspicious" and "Highly Suspicious" [22]. [22] first utilize a centroid clustering algorithm to split a large training dataset into several groups. Hereafter, a decision tree is employed to identify suspicious bidders. Social interactions between bidders and sellers in the online auction environment might be an effective sign of shilling behaviours. [20, 23] exploit a graph-based supervised learning approach to detect fraudsters. Both need to analyse enormous historical transactions. The approaches of machine learning could be divided into two major steps, i.e., (1) proposing a learning model to find outliers; (2) clustering outliers into several categories [17].

[24] applied genetic algorithms to detect optimally collusive behaviour. In this research, the fraud features, including social network analysis, economics of crime perspective, and original auction site's reputation mechanism, are transformed into fuzzy rules. Latent suspicious bidders are identified by those rules. Then, applying genetic algorithms to discover fraudsters. The highlight of their work might be the data collection, which contains the black account list from online auction site in Taiwan. The proposed approach might be easily verified.

[16] and [25] developed detection methods based on the Bayesian graph to calculate the probability to be a shill or not. To verify shill behaviours, four fraud patterns are adopted by [16] and seven by [25]. As a result of handling auction bids, the shill bidders are picked out.

2.2.2 Runtime Detection

In the following, we summarize the very few runtime fraud detection techniques [9], [11] and [2].

In [9], the authors developed a multi-agent trust management framework for the real-time detection of shills. User' sub-roles (five groups) are dynamically assigned according to the reputation and current shill scores of users. Yet, the reputation depending on the feedback ratings may be easily falsified. According to his sub-role, a user can be granted or denied auction services and resources. In this paper, each bidder is monitored with an auction agent, which may be not practical, especially for a large number of participants. During the auction, two actions can be taken in case of detected fraud: warning a suspicious bidder and cancelling the auction.

In [11], Xu et al. introduced a formal approach to detect shill bidding in live auctions. The approach employed three sources: the auction model which is updated dynamically as new bids arrive, Linear Temporal Logic (LTL) formulas representing the shill patterns, and a SPIN model checker that verifies whether the LTL formulas are violated or not. Nevertheless, monitoring an auction after every single submitted bid (the real-time event) may take the detection not efficient. This approach is also based on the estimated item price, which is not always possible for certain types of items, like antiques. Another deficiency of the above two approaches is the dubious feasibility and performance in real commercial online auction houses. Neither is based on large real auction data to conduct the experiment and report the detections.

Another interesting paper [2] presented a neural network based detection approach that classifies bidders into two groups, "normal" and "suspicious". To accurately cluster bidders, they normalize the shill bidding attributes by the weighted values. The classifier is initialized with a labelled training dataset, and then updated incrementally after each new bid coming. One of the difficult tasks in this study is labelling manually the bidders in the clusters that have been generated by a data clustering technique.

2.3 Commercial Auction Data

In the state-of-art, the evaluation of auction fraud detection systems is conducted with actual and/or simulated auction data. Actual auction data are significant in order to perform a robust empirical assessment because they represent the real behaviour of auction users. Numerous studies have extracted data from commercial auctions where the bid history can be accessed, like eBay, TradeMe and Yahoo! Taiwan. They developed their own Web scrapers that depend on the structure of the examined auction websites [26, 27]. Web scrapers extract raw data from web pages, and then convert them into usable information. Sometimes, the known list of fraudsters is employed as a starting point for the web crawlers. For instance, [28] examined manually several sources to identify ten fraudsters in eBay. [4] used the black-list of fraudsters that was provided by Yahoo! Taiwan, and [29] the suspended list of users that was released by Ruten of Taiwan. But in eBay these types of information (such as the blocked accounts) are not disclosed.

Nevertheless, the data extraction task is tedious and expensive as demonstrated in some papers. The tremendous volume of data in auction sites make the data crawling very difficult [26], and obtaining big data is a complicated task [27, 30]. Additionally, filtering the overwhelming amount of offline auction data is costly. [31] claimed that 80% of resources are used to pre-process that authentic auction data. To improve the time-efficiency, some studies developed concurrent crawler agents with multiple threads, and a queuing technique to avoid redundant crawling [26–28]. Even though, [27] scraped only 1300 auctions and 800,000 transactions from eBay after a period of 1 month. [26] extracted only 7682 auction pages from eBay after a period of 8 hours, but did not collect the bidding data. Also, we would like to mention that some commercial auctions provide some restrictions on crawling. As stated in the eBay policies, the use of scrapers to access its data is disallowed. Still, due to the data storage problem, auction sites delete data after a certain period. For example in eBay, this period is between 2 to 3 months depending on the item category. Therefore, the calculation of the fraud metrics is carried out on a small time period.

2.4 Conclusion

In this chapter, the related works for Shill Bidding (SB) are provided. Various SB strategies are introduced into several groups. From two perspectives, i.e., offline vs. runtime, the detection and prevention mechanisms for deterring auction frauds are presented. In addition, the discussion of the commercial auction dataset is proposed. Next chapter, our In-Auction Fraud Monitoring (IAFM) system is introduced.

Chapter 3

A Dynamic Run-Time Auction Fraud Monitoring System

This chapter contains three sections. Section 3.1 provides the top-level structure of the entire auction system and the communication among the various components. In Section 3.2, it describes our In-Auction Fraud Monitoring (IAFM) framework that we design as a dynamic Multi-Agent System (MAS). Finally, the detailed design of our system, including In-Auction Fraud (IAF) detection, reaction to IAF and cluster updating, are proposed in Section 3.3.

3.1 System Architecture

The entire auction system is organized with three independent layers as depicted in Figure 3.1. The UI layer is responsible for the interaction duties with the end users, like auction and user registration, bid placement and information display. The application layer is a MAS composed of two fixed agents (auction controller and user's cluster updating) and multiple dynamic IAF monitoring agents. The dynamic creation and deletion of the monitoring agents, the extraction and storage of auction and user data, the inspection of bidding activities, the reaction to fraud, and the revision of users' fraud scores are all performed in this layer. The data layer stores information about users, live and past auctions. The advantage of this 3-layer architectural style is mostly the easiness in maintaining each layer independently from others. In real-life, this architecture can be deployed on three main tiers: the auction website that we want to monitor, the IAFM tier, and the database server. The agents of IAFM may be distributed on several servers to increase the system scalability and reliability, since in practice hundreds of live auctions are operating in parallel.

We propose a controller-based MAS for monitoring IAF in runtime. The controller agent acts as a manager of the whole auction system and performs simultaneously several important operations: collecting auction data from multiple live auctions, creating multiple monitoring agents, and assuring the concurrent communication with them. We describe below the system work-flow and the interaction between the various components:

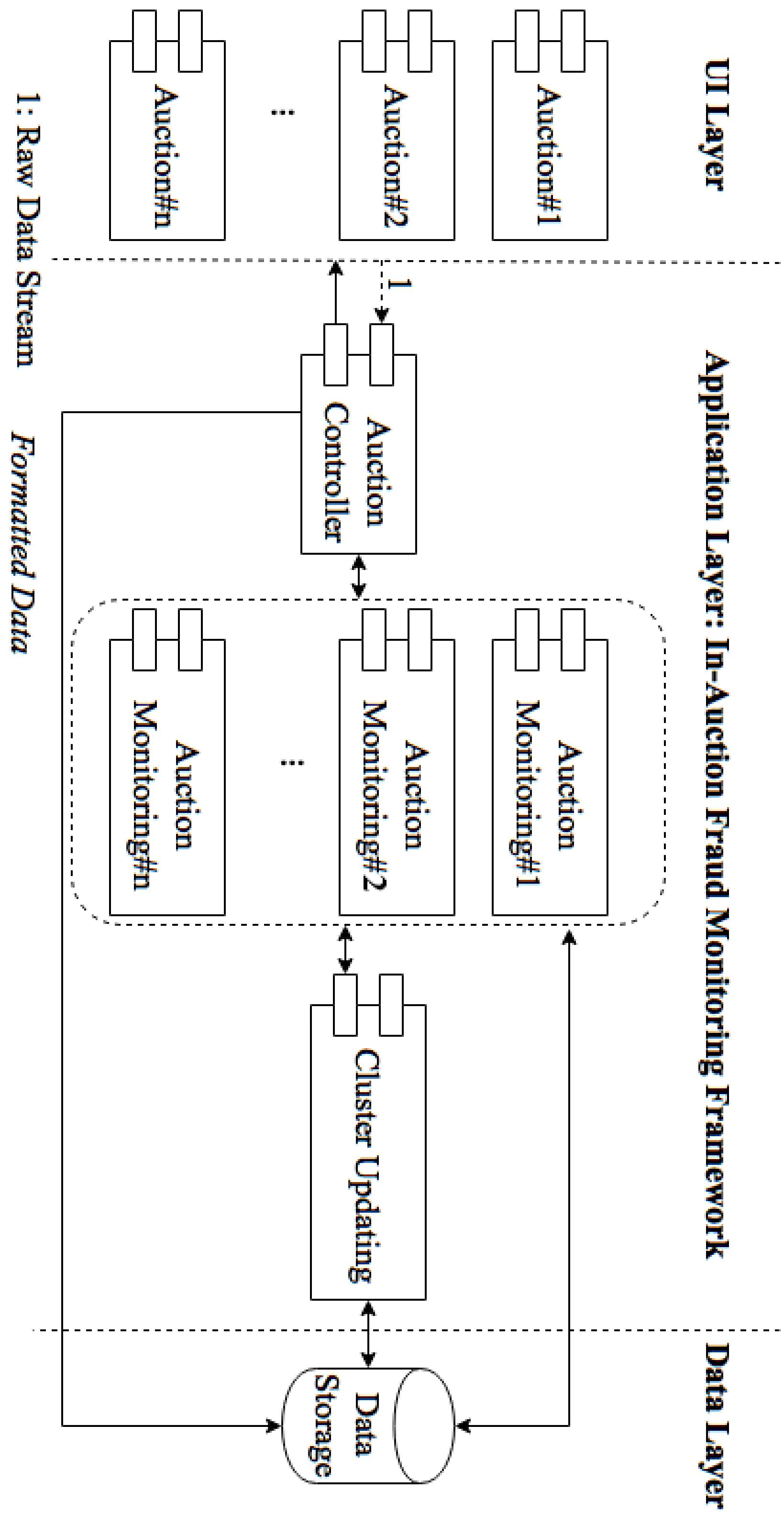

Figure 3.1: Auction System Architecture.

- *Dynamic Creation of Monitoring Agents:* When starting an auction, all its information should be first initialized, such as the category, description, starting and reserve prices of the item as well as starting time, duration and the bid increment rule of the auction. All this information along with the auction ID generated by AucController is then stored. Next, AucController creates a new AucMonitoring agent w.r.t the auction ID.

- *Collection of Live Auction Data:* AucController extracts continuously raw bidding data (like bidderID, bid price and bid time) from the ongoing auctions, formats and stores them in the right logs.

- *Stage-based Monitoring of IAF:* When one time point is reached in an auction, AucMonitoring agent fetches various information from the data store, evaluates them by using a set of IAF metrics and computes the bidders' live fraud scores. If suspicious and/or fraudulent bidding behaviours exist at this stage, AucMonitoring notifies AucController to warn the suspected bidders and/or terminate the corresponding auction. If an auction ends successfully, the auction site announces the winner.

- *Deconstruction of Monitoring Agents:* Every time an auction is finished successfully or not, before destroying itself, AucMonitoring triggers asynchronously ClusterUpdating agent to revise the participants' fraud scores and clusters. This self-destruction mechanism helps to reduce the workload on AucController.

- *Updating of Fraud Scores:* For each completed or cancelled auction, and for each of its participants, ClusterUpdating agent produces his new fraud value and cluster based on the past and live scores. If the updated cluster is fraudulent, the bidder's account is then suspended.

3.2 A Dynamic MAS

MASs yield to significant benefits [32], including: 1) an easy management of the system complexity by decomposing a challenging problem into sub-problems that are assigned to different agents; 2) an efficient computation when agents utilize asynchronous message passing, and therefore concurrent operations may be realized; 3) a great flexibility since agents may be added or removed easily from the society thanks to their autonomous feature. According to [33], agents enjoy the following characteristics: 1) Autonomy, i.e., agents perform actions based on their own knowledge without any external intervention; 2) Proactivity, i.e., agents take the initiative to adjust themselves to accomplish the predefined goals; 3) Reactivity, i.e., agents respond to the changes in their environments by taking suitable decisions; 4) Collaboration or social ability, i.e., agents communicate and coordinate with each others, typically by passing messages. Additionally, agents may be super agents, i.e., they are built with more capabilities (greater CPU power, huge storage capacity and higher network bandwidth) in order to perform huge workload [34]. Table 3.1 shows the characteristics of our three types of auction agents. We consider AucController and ClusterUpdatng as super agents with high processing power and network bandwidth since they interact concurrently with a large number of running auctions and monitoring agents. Since ClusterUpdating is triggered by AucMonitoring, it does not have the autonomy and pro-activity features.

We develop our IAMF as dynamic MAS because during runtime several auction monitoring agents can be created and destroyed and without disturbing other agents [35]. An adaptive configuration is a good approach to handle the scalability issue since in practice hundreds of auctions run in parallel. However, dynamic architectures need extra services to support them. When agents join and leave unpredictably the society, the agent location mechanisms are required in both centralized [36] and decentralized architectures [37, 38]. In other words, some agents in our IAMF need to know the

	AucController	**AucMonitoring**	**ClusterUpdating**
Autonomy	√	√	
Pro-activity	√	√	
Reactivity	√	√	√
Collaboration	√	√	√
Super-Agent	√		√

Table 3.1: Auction Agent Characteristics.

addresses of the agents they communicate with.

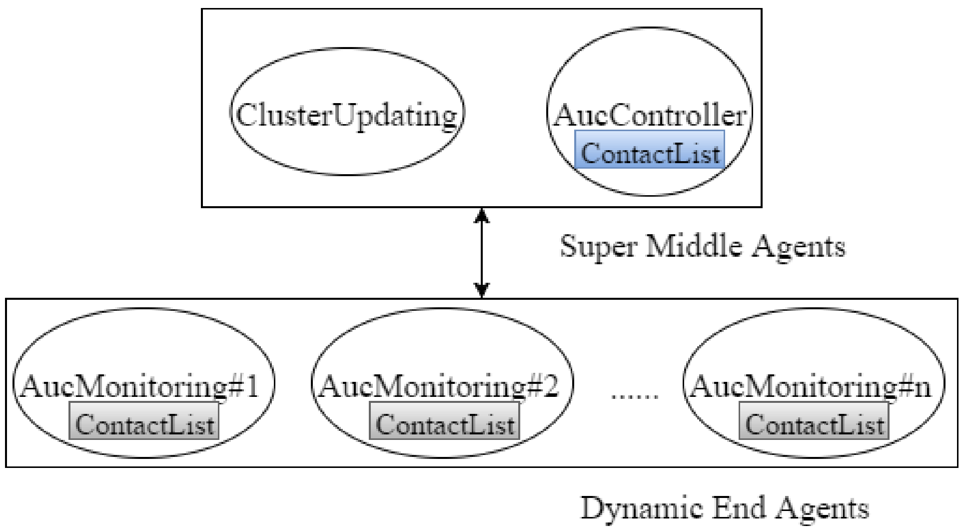

Figure 3.2: Agent Dynamic Location Mechanism.

We select the centralized middle-agent approach proposed by [36] to organize our IAFM service and make it much easier to develop and modify. An agent may have a contact list of the agents it interacts with, and this list may be changed dynamically. AucController, a middle agent, enables interactions with the dynamic end-agents by storing their location information in the community. As presented in Figure 3.2, AucController knows the addresses of the dynamic monitoring agents and the fixed agent ClusterUpdating as well. When a new monitoring agent is created, its contact list is initialized with the fixed location information of the two super-agent.

Also, the new location will be added to the contact list of AucController. With respect to ClusterUpdating agent, there is no need to equip it with a contact list. When a monitoring agent leaves the MAS, AucController deletes it from its list. The advantage of our design is that it is easily manageable since only the contact list of AucController is modified, and our system will have the ability to adjust itself to the internal and external changes of agents (such as their creation and destruction).

3.3 System Detailed Design

To increase trust in online auctioning, the live auction should be systematically monitored at different times. These time points are defined by the developers depending on the auction duration and IAF types. AucMonitoring consists of two internal agents as illustrated in Figure 3.3. At each auction time point, AucMonitoring initiates sequentially its internal agents, and then waits until the next time point is reached. When the current auction is completed or cancelled, AucMonitoring calls asynchronously ClusterUpdating agent to generate the bidders' new fraud scores. Subsequently, AucMonitoring automatically deletes itself from the MAS. This will help to release the MAS resources and keep it working more efficiently. The runtime fraud monitoring of one live auction is given in Algorithm 3.1, which uses asynchronous calls. This algorithm employs three logs: 1) live auction log contains various information such as auctionID, sellerID, productID, starting time, duration, starting and reserve prices, bid increment, submitted bids of each participant, and his current live fraud score; 2) when an auction is completed or terminated, all its information are transferred from the live auction log to the auction history log by including new data, like the final price and total bids according to the stage reached in the auction, winnerID in case the auction was successful, and auction status (successful or unsuccessful); 3) the fraud pattern log records the fraud metrics, their corresponding auction time point and thresholds. To take into account new fraud patterns, we just add their metrics into this log.

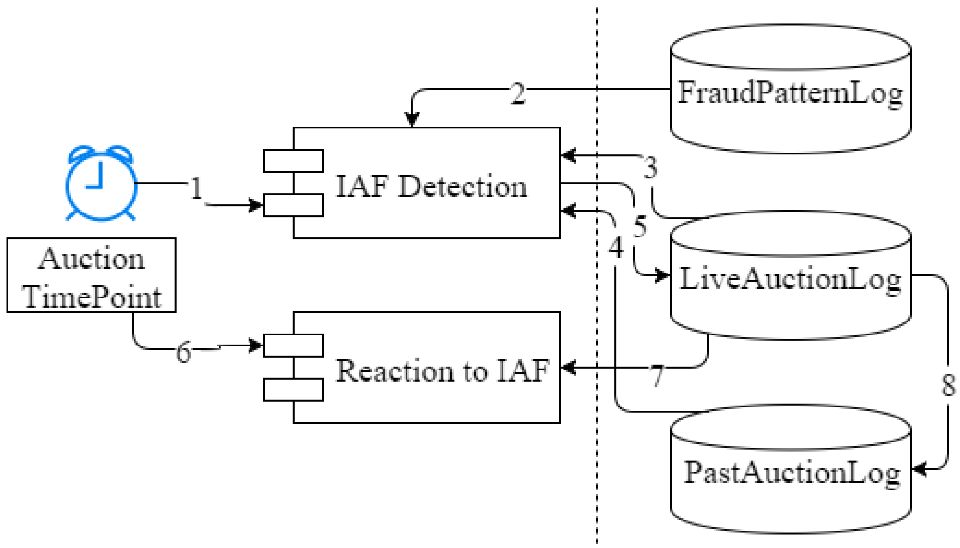

Figure 3.3: Monitoring Agent of Live Auction.

Algorithm 3.1 MonitorLiveAuction

Input: $\mathrm{a_i}$ //*live auction*
Sources: LiveAuctionLog, PastAuctionLog
$time \in \{T_1, T_2, , T_\mathrm{n}\}$;
// *time points defined w.r.t to IAF type and auction duration*
time = T_1;
while $((status_{\mathrm{ai}} = \text{“}successful\text{”})and(time <= T_\mathrm{n}))$ **do**
    // *monitor as long as the auction was not cancelled nor completed*
    detectInAuctionFraud($\mathrm{a_i}$, time);
    // *compute liveFraudScore for each bidder in auction a_i at time point*
    reactToInAuctionFraud($\mathrm{a_i}$);
    // *analyze liveFraudScore to take actions*
    // *asynchronous call*
    time = detectNextTimePoint();
    // *wait until the next time point is reached*
end while
ClusterUpdating($\mathrm{a_i}$);
// *asynchronous call*
transfer($\mathrm{a_i}$, PastAuctionLog);
selfdestroy();

3.3.1 Runtime Detection of IAF

A bidder cannot be blamed with only few fraud patterns but a series of patterns should be used to improve the detect accuracy in each ongoing auction. We monitor each bidder as he can change his behaviour in just few auctions. We consider properties from various sources such as from users, auctions and bids. Each IAF pattern is most likely to occur in a certain auction stage. To generate more accurately the users' live fraud scores, the detection algorithm below (shown in Algorithm 3.2) evaluates a stage-based set of IAF patterns w.r.t the data of bidder being examined, the auction being monitored and the submitted bids. We first quantify the fraud patterns with proper metrics and then aggregate them with a fusing method, such as the weighted mathematical average. Indeed, we can associate weights to IAF patterns to denote their relative importance. These metrics are computed from the past and/or progressing auctions. The aggregation goal is to produce an overall value, called live Fraud Score, which measures the level of fraud of each bidder in each running auction. The live fraud score is updated at each auction time point by including the set of fraud metrics of the corresponding stage.

Algorithm 3.2 detectInAuctionFraud

Input: $\mathrm{a_i}$ *//live auction*, time *//time point*
Sources: LiveAuctionLog, PastAuctionLog, FraudPatternLog

setofMetrics = deployFraudMetrics(time);
for $u_j \in a_i$ **do**
    setOfData = extractData($\mathrm{a_i}$, $\mathrm{u_j}$, time);
    // collect data of users, auctions and submitted bids
    $\mathrm{stageScore_{Uj}}$=0;
    for $m \in setOfMetrics$ **do**
        scoreMetric = evaluate(m, setOfData);
        // evaluate a fraud metric with data of user u_j, auction $\mathrm{a_i}$ and bids
        $\mathrm{stageScore_{Uj}}$ = mergeScores1($\mathrm{stageScore_{Uj}}$, scoreMetric);
        // combine the fraud metrics of an auction stage
    end for
    $\mathrm{liveFraudScore_{Uj}}$ = mergeScore2($\mathrm{liveFraudScore_{Uj}}$, $\mathrm{stageScore_{Uj}}$);
end for

3.3.2 Runtime Reaction to IAF

The fraud reaction agent reacts to a bidders' behaviour upon the value of his current live fraud score at each auction time point in each progressing auction. If this score is in a certain range or beyond it, this agent performs immediately the following actions.

- *Warn suspicious bidders:* This agent sends a warning message to AucController about suspicious bidders in the correspondent auction. AucController warns the suspected bidders to bid more responsibly in the auction.

- *Cancel an auction:* If serious cheating activities exist, the agent sends a cancellation message to AucMonitoring to terminate the infected auction. Next, it updates liveAuctionLog with the new auction status, i.e., "unsuccessful".

- *Blame fraudulent bidders:* For all high live fraud scores, AucController will also contact the fraudulent bidders and blame them for the auction termination.

We may note that in the same auction, we may detect several suspicious and/or fraudulent bidders. In Algorithm 3.3, the two thresholds are defined by the developer.

Algorithm 3.3 reactToInAuctionFraud

Input: $\mathrm{a_i}$ //*live auction*
Sources: LiveAuctionLog
for $u_\mathrm{j} \in a_\mathrm{i}$ **do**
  if $(liveFraudScore_\mathrm{Uj} \in highRange)$ **then**
    cancelAuction($\mathrm{a_i}$);
    $\mathrm{status_{ai}}$ = "unsuccessful";
    break;
  end if
end for
for $u_\mathrm{j} \in a_\mathrm{i}$ **do**
  if $(liveFraudScore_\mathrm{Uj} \in highRange)$ **then**
    blame("Fraudulent", $\mathrm{u_j}$,$\mathrm{a_i}$);
  else if $(liveFraudScore_\mathrm{Uj} \in medRange)$ **then**
    warn("Suspicious", $\mathrm{u_j}$, $\mathrm{a_i}$);
  end if
end for

3.3.3 Users' Cluster Updating

When an auction ends successfully or unsuccessfully, the current live fraud score of each participant is merged with his fraud score of the past auctions, and his cluster is updated accordingly (cf. Algorithm 3.4). Hence, we obtain an overall value representing the fraud score in all the participated auctions. We give a chance to bidders to improve their behaviours. It is useful to assign to each user a cluster regarding his level of conduct: "normal", "suspicious", or "fraudulent". In this way, we will have a very good idea about each user's behaviour in the auction community. The cluster of each bidder is displayed to all to see, and could represent his reputation. A new user will have a status of "normal". Only normal and suspicious bidders can negotiate in the new auctions. The accounts of fraudsters are suspended permanently.

Algorithm 3.4 updateUserFraudScoreAndCluster

Input: $\mathrm{a_i}$ //*live auction*
Sources: LiveAuctionLog, UserLog

```
for u_j ∈ a_i do
    fraudScore_Uj = mergeScore3(fraudScore_Uj, liveFraudScore_Uj);
    if (fraudScore_Uj ∈ medRange) then
        cluster_Uj="suspicious";
    else if (fraudScore_Uj ∈ highRange) then
        cluster_Uj="fraudulent";
        suspendAccount(u_j);
    end if
end for
```

3.4 Conclusion

In this chapter, the IAFM architecture and detailed design have been explained. The dynamic features of IAFM and agents' responsibilities are illustrated accordingly. Four algorithms, including monitoring, detecting, reacting, and updating, are described. In Chapter 4, the implementation with shill bidding is presented, which contains shill bidding patterns and computation mechanisms. Moreover, the implementation overview of IAFM is shown in the final.

Chapter 4

Implementation with Shill Bidding

This chapter presents the implementation with shill bidding. First of all, in Section 4.1, we provide the set of shill bidding patterns that we use in each auction stage. Secondly, the metrics for computing shill scores are presented in Section 4.2. Section 4.3 illustrates the values of weights and computation mechanisms. Finally, the Section 4.4 displays the implementation of our IAFM.

4.1 Stage-based Shill Bidding Patterns

Many shilling strategies have been discovered in past studies. Below we analyzed and compiled 17 shill patterns from [7, 8, 10, 11, 13–16]. Properties from users, bids and auctions may all be taken into account to ensure better detection results. Patterns #1, #10 and #12 are for the users; #15, #16, and #17 for the auctions, and the rest is for the submitted bids. Auction patterns means that the auction involves fraudsters. We monitor shill bidding at three auction time points: at 25%, at 90% and at 100% of the auction duration. In the following, we determine for each of them the shill patterns that most probably occur at the corresponding stage.

Stage[$\mathbf{T_{start}}$, $\mathbf{T_{early}}$]:

1. SB participates exclusively in auctions conducted by some sellers. A normal bidder may negotiate in several concurrent auctions to find the best price, but a SB deals with a limited range of sellers (**Bidder Tendency**). Concurrent auctions mean that they sell identical items.
2. SB places bids very close to the auction starting time (**Early Bidding**).
3. SB submits a bid that is very close to the reserve price.
4. SB posts small bid increments with the minimum amount required by the auction.

Stage[$\mathbf{T_{early}}$, $\mathbf{T_{middle}}$]:

5. SB outbids legitimate bids until he is satisfied or he has reached the reserve price (**Bidding Ratio**).
6. SB often bids successively to outbid oneself even when he is the current winner (**Successive outbidding**).
7. Successive outbidding and bidding ratio are high when the current auction price is smaller than the reserve price; otherwise they are lower to reduce the risk of winning (**Reserve Price Shilling**).
8. SB submits a bid within a short time interval (1 minute) of any new legitimate bids.
9. SB outbids any bid with a minimum of 10% to 20% of the current bidding price.
10. SB participates in concurrent auctions with higher bidding prices rather than with lower prices.

Stage[$\mathbf{T_{middle}}$, $\mathbf{T_{final}}$]:

11. SB stops negotiating early before the auction ends i.e., avoids sniping (**Last Bidding**).

12. Winner ratio of SB in past auctions is very low even when his bids aggressively (**Winning Ratio**).

13. SB bid less for high or medium value items.

14. SB submits low bid increment with the minimum amount required by the auction.

15. An auction with shills has more bids that the average number of bids in normal concurrent auctions (**Auction Bids**).

16. Starting price of an infected auction is less than the average staring price of concurrent auctions.

17. When the auction price is significantly higher than the expected price, there is a probability of 66.7% of the auction being infected [10].

4.2 Shill Bidding Metrics

We defined here the metrics for eight shill patterns (those shown in bold) to show the feasibility of our IAF monitoring approach. The metrics are calculated from currently examined auctions and/or offline auctions covering a certain period of time. The higher the metric value, the more suspicious the observed bidder is.

1. **Bidder Tendency:**

$$\begin{aligned}
&\textbf{if } (|auctionPart(u_j)| > 1)\textbf{then} \\
&\qquad bidderTendency(u_j, s_k) = \frac{|auctionSeller(u_j, s_k)|}{|auctionPart(u_j)|} \\
&\textbf{else } bidderTendency(u_j, s_k) = 0
\end{aligned}$$

where the dividend is the number of auctions that user $\mathrm{u_j}$ has participated in for seller $\mathrm{s_k}$ in a given time period; the divisor is the total number of auctions that user $\mathrm{u_j}$ has joined during the same period. The condition is necessary to discard those bidders who participated in only one auction. We may note that both operands include the currently observed live auction.

2. **Early Bidding:**

$$earlyBidding(u_{\mathrm{j}}, a_{\mathrm{i}}) = 1 - \frac{firstBidTime(u_{\mathrm{j}}, a_{\mathrm{i}}) - startTime_{\mathrm{ai}}}{duration_{\mathrm{ai}}}$$

3. **Bidding Ratio:**

$$biddingRatio(u_{\mathrm{j}}, a_{\mathrm{i}}) = \frac{totalBids(u_{\mathrm{j}}, a_{\mathrm{i}})}{totalBids(a_{\mathrm{i}})}.$$

For both operands, the number of bids should be collected only from the stage $[\mathrm{T_{early}}, \mathrm{T_{middle}}]$.

4. **Successive Outbidding:**

$sucBid = 0;$

if $(succOutbid(u_j, a_i, 3))$**then** $succBid(u_j, a_i) = 1;$

else if $(succOutbid(u_j, a_i, 2))$**then** $succBid(u_j, a_i) = 0.5;$

If user $\mathrm{u_j}$ successively outbids two or three times in the stage $[\mathrm{T_{early}}, \mathrm{T_{middle}}]$, the value of this fraud pattern is 0.5 or 1 respectively. Otherwise, the value equals 0.

5. **Reserve Price Shilling:**

$$\textbf{if}(auctionPrice(a_i) < reservePrice_{\mathrm{ai}})\ \textbf{then}$$

$$\textbf{if}(succBid(u_j, a_i) >= 0.5) or (biddingRatio(u_j, a_i) > 0.5)\ \textbf{then}$$

$$reservePriceShill(u_j, a_i) = 1$$

$$\textbf{else}\ reservePriceShill(u_j, a_i) = 0$$

where biddingRatio is computed from the interval [$\mathrm{T}_{\mathrm{early}}$, $\mathrm{T}_{\mathrm{middle}}$].

6. **Last Bidding [Xu2010]:**

$$lastBidding(u_j, a_i) = \frac{endTime(a_i) - lastBidTime(u_j, a_i)}{duration_{\mathrm{ai}}}$$

7. **Winning Ratio:**

$$winningRatio(u_j) = 1 - \frac{|auctionWon(u_j)|}{|auctionPartHigh(u_j)|}$$

$$auctionPartHigh(u_j) = \{a_i | biddingRatio(u_j, a_i) > 0.05\}$$

The dividend is the number of auctions won by user $\mathrm{u_j}$; the divisor is the number of auctions joined by user $\mathrm{u_j}$ and in these auctions $\mathrm{u_j}$ has a high bidding ratio. This will eliminate the issue of non-active bidders. Here *biddingRatio* is calculated from the whole auction interval, i.e., [$\mathrm{T}_{\mathrm{start}}$, $\mathrm{T}_{\mathrm{final}}$]. Again, both operands include the currently observed live auction since it is already completed.

8. **Auction Bids:**

$$\textbf{if}(averageBids(a_i) < totalBids(a_i))\ \textbf{then}$$
$$auctionBids(a_i) = 1 - \frac{averageBids(a_i)}{totalBids(a_i)}$$
$$\textbf{else}\ auctionBids(a_i) = 0$$

where *averageBids* is generated from the concurrent auctions to auction a_i for a certain time period, and totalBids is produced from the whole auction, i.e., $[T_{start}, T_{final}]$.

4.3 Stage-based Shill Bidding Detection

Several works have assigned manually weights to their shill bidding patterns [6, 8, 22]. We follow the same approach as explained below (see Table 4.1):

- The IAF patterns at the early stage have the lowest weights because we need more fraud signs to take actions. In particular, Bidder Tendency pattern has a low weight due to the false tendency issue, i.e., a bidder has a tendency for a certain seller due to his good reputation, or he is the only one selling the item [8].
- In the middle stage, the corresponding IAF patterns have the highest weights because they are a good indicator of shilling (a shill bidder bids aggressively in this stage). Still, we have some exceptions. The reserve price shilling has a low weight because in commercial auctions, the reserve price (which is hidden from bidders) is not accessible. In our experiments, we generated artificially the values of this attribute.
- At the final stage, the IAF patterns have medium weights. However, since

Winning Ratio represents the behaviour of a user in all the past auctions, thus we give it a high weight. Also, we assign a low weight to Auction Bids because it is a property of auctions.

In Table 4.2, we show how to compute the live fraud score at each auction time point for each bidder in each ongoing auction. The live score is updated by including the fraud metrics that are computed for the current auction stage. This updating is based on the weighted mathematical average.

Stages	**Priority**	**Fraud Patterns**	**Fraud Category**	**Weights**
Early Stage	Low	Early Bidding	Bid Property	0.3
		Bidder Tendency	User Property	0.3
Middle Stage	High	Bidding Ratio	Bid Property	0.8
		Successive Outbidding	Bid Property	0.8
		Reserve Price Shilling	Bid Property	0.3
Final Stage	Medium	Last Bidding	Bid Property	0.5
		Winning Ratio	User Property	0.8
		Auction Bids	Auction Property	0.3

Table 4.1: IAF Weights.

$\mathbf{T_{early}}$	$\mathbf{T_{middle}}$	$\mathbf{T_{final}}$
$W_{\text{early}} = W_{\text{EB}} + W_{\text{BT}}$	$W_{\text{middle}} = W_{\text{BR}} + W_{\text{SO}} + W_{\text{RPS}}$	$W_{\text{final}} = W_{\text{LB}} + W_{\text{WR}} + W_{\text{AB}}$
$S_{\text{early}} = W_{\text{EB}} * EB + W_{\text{BT}} * BT$	$S_{\text{middle}} = W_{\text{BR}} * BR + W_{\text{SO}} * SO + W_{\text{RPS}} * RPS$	$S_{\text{final}} = W_{\text{LB}} * LB + W_{\text{WR}} * WR + W_{\text{AB}} * AB$
$LFS_{\text{early}} = \frac{S_{\text{early}}}{W_{\text{early}}}$	$LFS_{\text{middle}} = \frac{S_{\text{early}} + S_{\text{middle}}}{W_{\text{early}} + W_{\text{middle}}}$	$LFS_{\text{final}} = \frac{S_{\text{early}} + S_{\text{middle}} + S_{\text{final}}}{W_{\text{early}} + W_{\text{middle}} + W_{\text{final}}}$

Table 4.2: Bidders' Live Fraud Score at each Auction Stage.

4.4 Overview of Implementation

We developed our In-Auction Fraud Monitoring (IAFM) system with the Belief-Desire-Intention (BDI) model [39] [40]. Each agent has a set of beliefs (agent's knowledge about itself and its environment), goals (the desires an agent intends to achieve), plans (the assigned tasks an agent performs). As you can see in Figure 4.1, it shows the plans within the multi-agent community and we implemented 9 plans and 15 classes. The source code of three BDI agents, i.e., AucController, AucMonitoring, ClusterUpdating, are shown in Appendix A.

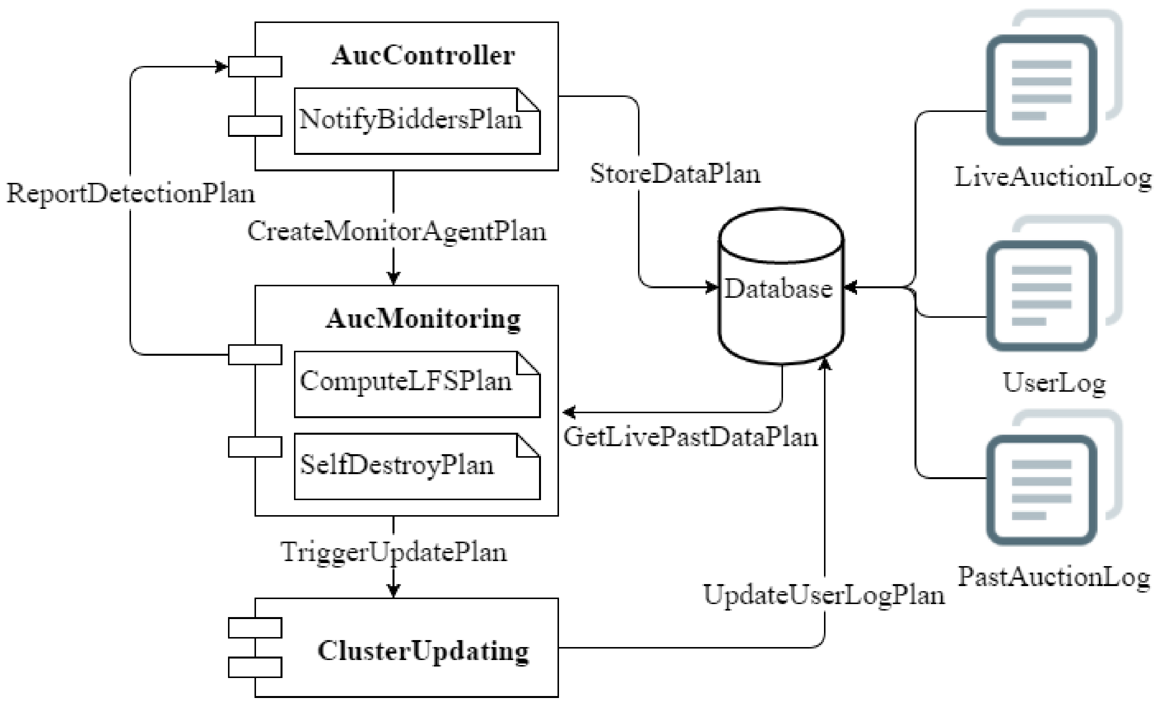

Figure 4.1: Agent and Plan Implementation.

The particular role of each plan is described below:

1. **AucController**

    - StoreDataPlan: This plan is responsible for extracting as well as formatting raw bidding data from the running auctions, and then storing them into the database.

- CreateMonitorAgentPlan: When one auction starts, *CreateMonitorAgent-Plan* creates a new AucMonitoring agent corresponding to the running auction.

- NotifyBiddersPlan: An auction status contain "successful" or "unsuccessful". When the auction ends successfully, *NotifyBiddersPlan* announces the winner and the winning price. Otherwise, it sends warnings to suspicious bidders or addresses the fraudulent bidders.

2. **AucMonitoring**

- GetLivePastDataPlan: This plan extracts live and past auction data as well as users' profiles from the database.

- ComputerLFSPlan: *ComputeLFSPlan* applies them to compute the live fraud score of each participant in each live auction.

- ReportDetectionPlan: After obtaining the live fraud score of each bidder, *ReportDetectionPlan* analyzes them by comparing with certain thresholds. According Algorithm 3.3, the actions are decided and sent to **AucController** agent.

- TriggerUpdatePlan: When the monitoring work is finished, *TriggerUpdatePlan* activates **ClusterUpdating** agent.

- SelfDestroyPlan: **AucMonitoring**agent automatically leaves the monitoring system by adopting *SelfDestroyPlan*.

3. **ClusterUpdating**

- UpdateUserLogPlan: According to Algorithm 3.4, *UpdateUserLogPlan* merges current and past fraud score to obtain the fraud score. Then, it updates the fraud score of each bidder in the user log.

For the purpose of simulation, we adopted the agent platform Jadex [41] that utilizes the BDI model as the reasoning engine as well as the FIPA-ACL as the agent interaction protocol. We employed the latest version Jadex BDI V3, and also two integrated development tools Eclipse IDE 4.4.1 and Java SE Runtime Environment 8u51. As for the database, we chose MySQL community Server. Within Jadex, we implemented all the agents, including beliefs, plans, and goals, in a set of Java classes. In addition, pure Java classes are transformed into BDI agents or Goals, and the fields and member functions into beliefs and plans [42]. Jadex supports the asynchronous communication through the public interface "IFuture" of Java. Also, the annotation "@Agent" and "BDIAgent" class are provided for creating agent easily. To remove an agent from the system, we apply the public function "agentKilled()" of an interface "IMicroAgent". In our work, we specifically implemented our In-Auction Fraud Monitoring (IAFM) to detect and react to the shill bidding strategies. Figure 4.2 illustrates the Jadex Control Center. Figure 4.3 and Figure 4.4 present the bidding information and monitoring result of the first experiment of Section 5.2.1.

4.5 Conclusion

This chapter expounds three dominating parts, i.e., fraud patterns, metrics, and computation mechanism. At last, the implementation of IAFM are displayed. Chapter 5 presents the system validation based on three conducted experiments. Two commercial auction datasets from eBay are selected.

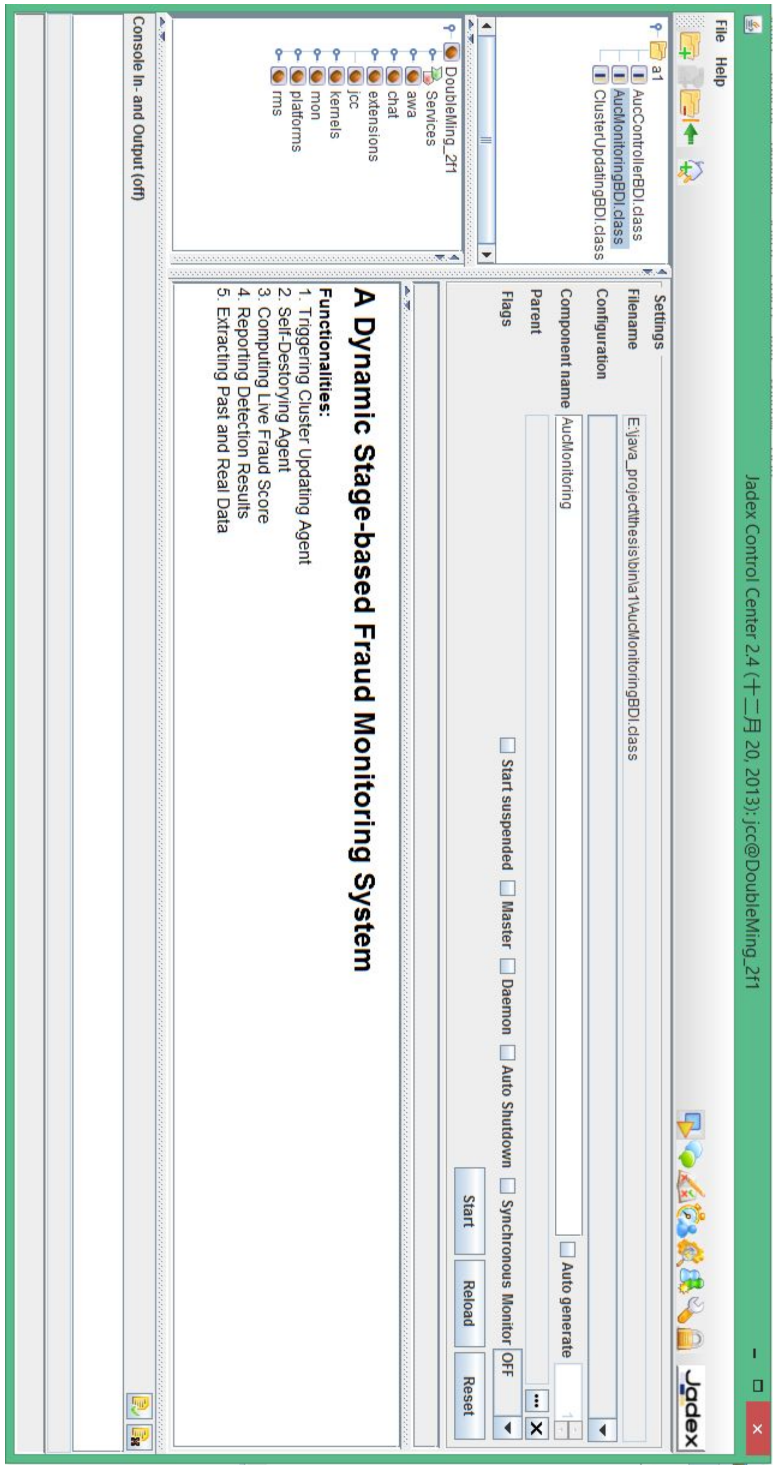

Figure 4.2: The Implementation Overview.

Bidding Details of Auction

AuctionID	BidAmt	BidTime	BidderID
3018788243	0.5	0.098136574	t***n
3018788243	5	0.511516204	t***6
3018788243	0.62	0.5978125	t***n
3018788243	1	0.597928241	t***n
3018788243	5	0.598078704	t***n
3018788243	10	0.59818287	t***n
3018788243	30	0.630231481	m***m
3018788243	12	1.199212963	t***n
3018788243	15	1.199375	t***n
3018788243	20	1.199513889	t***n
3018788243	25	1.199722222	t***n
3018788243	38.88	1.211747685	k***0
3018788243	50	1.218449074	m***m
3018788243	50	2.729722222	t***n
3018788243	51	3.708993056	f***e

Figure 4.3: The Bidding Information.

Montoring Agent Detection Result

BidderID	LFSearly	LFSmiddle	LFSend	Cluster
t***n	0.493	0.0986	0.0643	Normal
t***6	0.9635	0.1927	0.1257	Fraudulent
m***m	0.455	0.6998	0.5621	Suspicious
k***0	0.9134	0.1827	0.1911	Fraudulent
f***6	0.0	0.0376	0.0205	Normal
l***r	0.0	0.0188	0.0102	Normal
p***4	0.0	0.4137	0.2698	Normal
h***a	0.0	0.3176	0.1726	Normal
m***m	0.0	0.0188	0.1537	Normal
c***9	0.0	0.0	0.1699	Normal
r***l	0.0	0.0	0.1622	Normal
t***t	0.0	0.0	0.1761	Normal
r***7	0.0	0.0	0.1107	Normal
k***6	0.0	0.0	0.1107	Normal

Figure 4.4: The Monitoring Result.

Chapter 5

System Validation

In this chapter, we discuss three experiments to assess our auction monitoring system based on actual data from commercial auction house eBay. Section 5.1 exposes the statistical information of the real auction datasets that used in our experiments. Section 5.2 and Section 5.3 present the monitoring result and anlysis of two PDA auctions and one Xbox auction. In the end, the discussions of our experiment is explained in Section 5.4.

5.1 Real Auction Datasets

We have utilized real auction data that have been made available in the following link: http://www.modelingonlineauctions.com/datasets. This website contains numerous auction listings (English, forward, and one unit of each item) of three high-value items auctioned in eBay: XBOX game consoles, Cartier wristwatches, and Palm PDAs. For our experiments, we selected two items, Palm PDA and XBOX, for the following reasons. These items were in high demand as they attracted a large number of bidders and bids. Also, according to eBay website, today XBOX is in the top 2 of the most sold categories (among 34), and Palm PDAs in the top 12. Moreover since these items have good price ranges, they may have attracted fraudsters. In fact, more the item price is high, more there is a possibility of shill bidding activities [19]. The completed Palm PDA and XBOX auctions were collected over a period of two months in different years, 2003 and 2007 respectively.

	Features	**Comments**
Auction Aspects	auctionID	Unique identifier of an auction
	sellerID	Unique identifier of a seller
	opendBid	Starting price set by a seller
	price	Final price of an auction
	duration	7 days
	startTime	Initialized to 0
	endTime	startTime+duration
Bid History	bidID	Bid placed by a bidder/proxy bid
	bidTime	Time of the placement of a bid
	bidder	Unique identifier of a user
	bidderRate	Feedback rating of a bidder

Table 5.1: Auction Features.

Table 5.1 exposes the auction features, but we may note that the sellerID is missing in the dataset of XBOX. Table 5.2 shows some statistical information. In eBay, each auction user is uniquely identified. In Table 5.2, “1-time bidder” means that he participated in only one auction, and “2-time bidder” in only two auctions. We may note that the reserve price (which is hidden from bidders) is not accessible in eBay. Since this feature is required to compute the reserve price shilling metric, we therefore added it artificially into all the auctions (with a total of 242). We produced the reserve price for each auction according to its final price. Usually, on average, there is a difference of 15% between the final price and the reserve price [43]. It worth nothing that several papers used both original and crafted data to evaluate their reputation system for commercial online auctions [3, 44].

	PDA	Xbox
Total Auctions	149	93
Total Sellers	71	NA
Total Bidders	1024	656
Total Bids	3166	1861
Avg. Total Bidders	7	7
Avg. Total Bids	21	20
Avg. Winning Price	229.04	134.6
1-Time Bidders	75.2%	86.1%
2-Time Bidders	14.3%	10.4%

Table 5.2: Dataset Statistical Information.

5.2 Monitoring Palms PDA Auctions

To better understand how does our IAFM work, we present the Figure 5.1. Two live auctions are running parallelly. AucController creates two AucMonitoring agents for taking care of the bids in each auction. If shill bidding behaviours happen in the auction, AucMonitoring will notify the AucController with specific actions. After the auction is over, AucMonitoring will trigger the ClusterUpdating to update fraud score of each bidder participating in the auction.

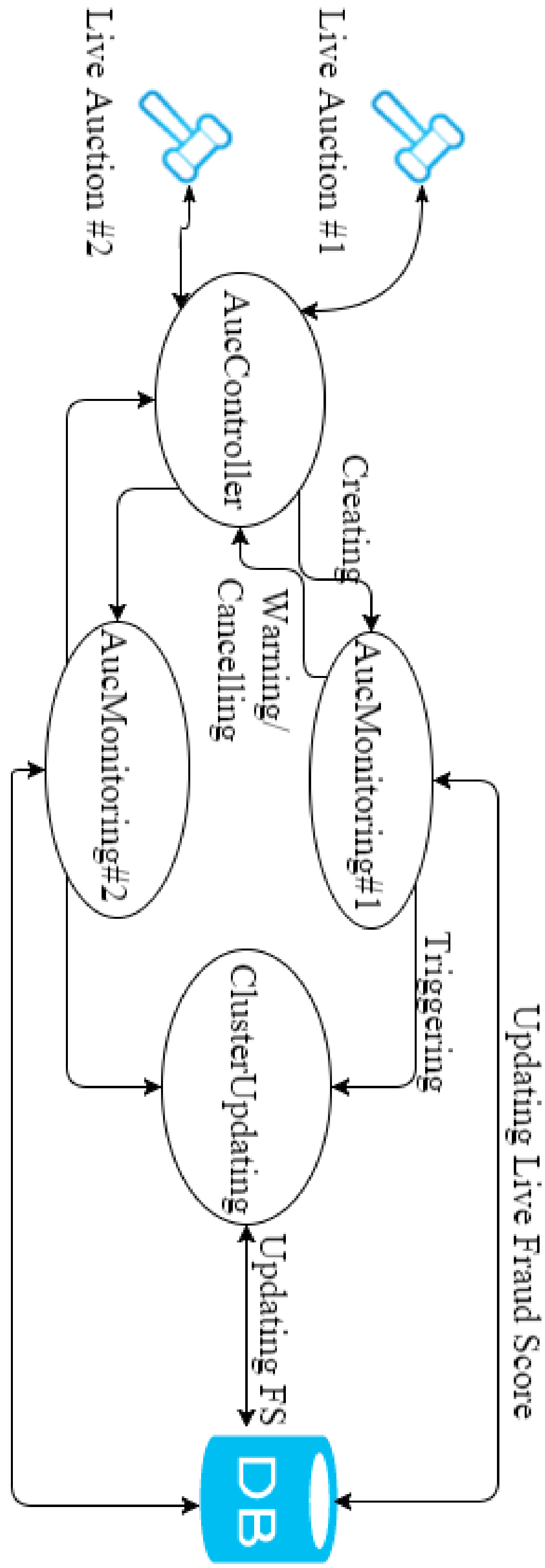

Figure 5.1: IAFM Monitoring Multiple Auctions.

As presented in Figure 5.2, the number of bids per auction is pretty high in the PDA dataset. Since shill bidding happens in auctions with more bids, we consider as our live auctions the top 2 auctions that have the highest number of bids. Hence, Our IAF metrics will be computed from those 2 top auctions (as the live auctions) as well as from the 147 remaining auctions (as the past auctions). We see in Figure 5.3 that most of the sellers held less then 5 auctions, and there is one particular seller (with the sellerID of "s***l") who held the highest number of auctions (40). It is this seller who launched the top two auctions. To protect the privacy of users, we return the userID with the first and last characters with three "*" inside. The duration of the examined auctions is 7 days. Therefore, the early stage takes 1 day 18 hours, the middle stage 4 days and 13.2 hours, and the final stage 16.8 hours. To monitor shill bidding activities in the selected auctions, we set the thresholds *medRange* to $[0.5, 0.7)$ and *highRange* to $[0.7, 1.0]$.

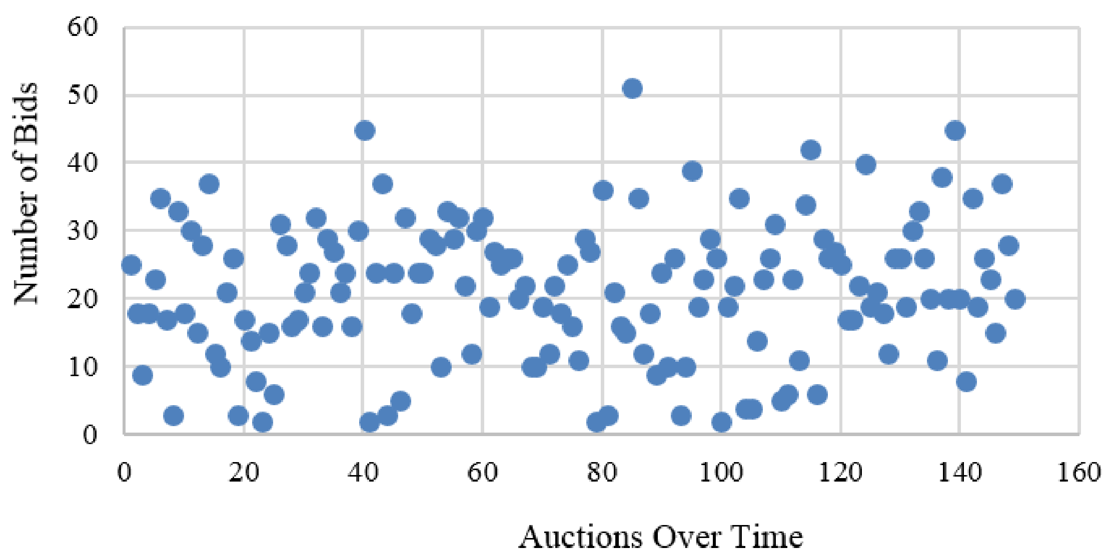

Figure 5.2: Bid Distribution in PDA Dataset.

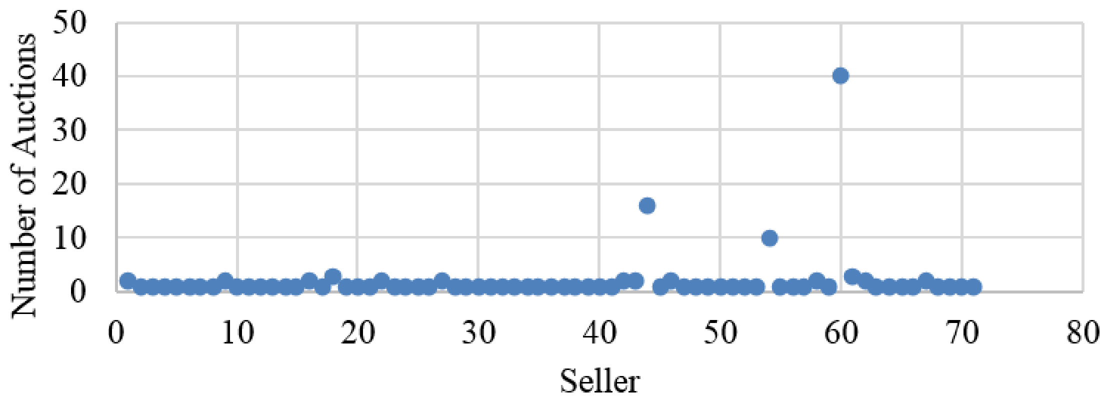

Figure 5.3: Seller Distribution in PDA Dataset.

5.2.1 Auction #1

Table 5.3 exposes the information of the first top auction. "1-time bid" means here the percentage of bidders who placed one bid in an auction, and "2-time bid" for two bids. Figure 5.4 gives the live fraud score of each bidder in each stage of this auction. Some bidders only have one stage live fraud score because of no bids placed by them in the other stages. For example, "r***l" only participated in the last stage, therefore the live fraud scores of the first two stages are zero. According to Figure 5.4, we have one suspicious and two fraudulent bidders. Consequently, in Table 5.4, we analyzed in detail the bidding activities of these bidders to confirm their shill bidding behaviours. In the early stage, "t***6" and "k***0" placed bids in the very beginning to attract more bidders, and both bided for only one seller. Regarding "m***m", he competed aggressively in the middle stage, but in the final stage, he stopped bidding very early. Besides he never won any auction. These behaviours are strong signs of fraud. We also computed the percentage of bidders who committed each IAF in this auction. As an example, the bidder percentage for early bidding is 18.2%, bidding ratio in the middle stage is 43.2%, and last bidding is 38.6%.

Palm Pilot M515 PDA (2013(4/14 4/20))-AuctionID: 3018788243			
Auction Details		Statistics (%)	
Starting price	$0.01	1-time bid	42.8%
Total bids	44	2-time bid	35.7%
Total bidders	14	Early stage	29.5%
Reserve price	$208.25	Middle stage	38.7%
Winning price	$245	Final stage	31.8%

Table 5.3: Auction #1 Statistical Information.

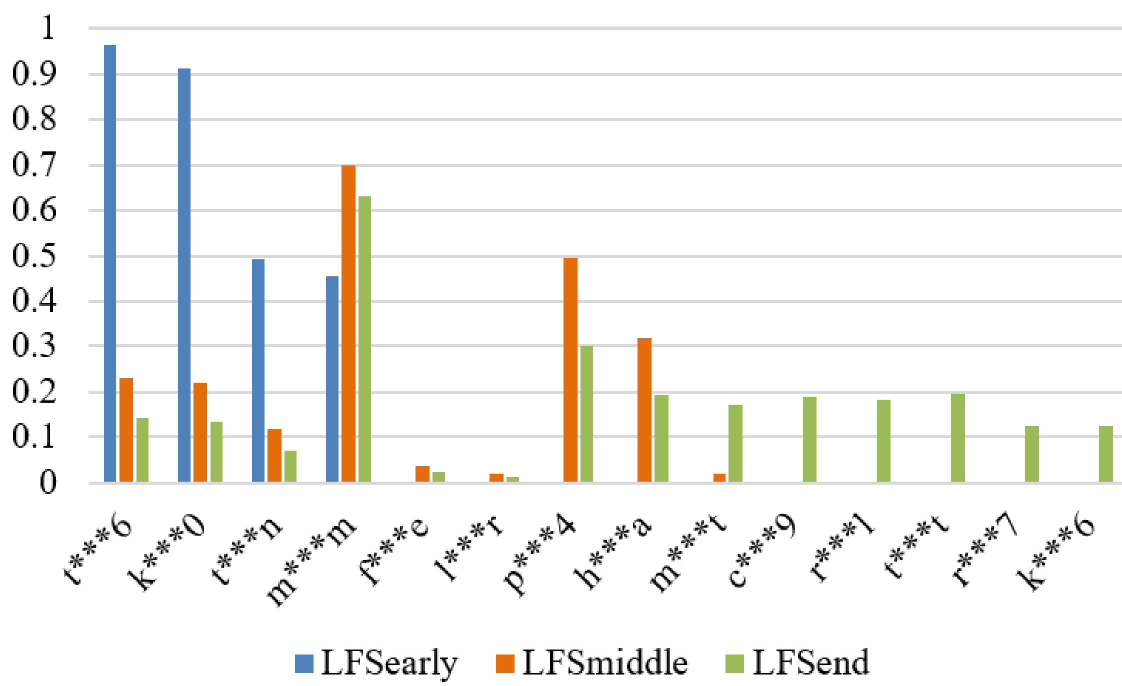

Figure 5.4: Runtime Detection in Auction #1.

According to our IAFM system, the following actions should have been taken against the detected shill bidders are shown in Table 5.5. Actually, this auction should have been cancelled at the early stage since there is a presence of two bidders with a very high IAF score. After the auction termination, the clusters of "t***6" and "k***0" are revised to "fraudulent" by considering their past fraud scores and live scores. Therefore, their accounts will be suspended. The cluster of "m***m" will be labelled as “suspicious”.

5.2.2 Auction #2

This experiment presents the monitoring results of the second auction. Table 5.6 presents the detailed information of this auction, and Figure 5.5 the live fraud scores of each bidder in each stage. In this auction, only one bidder, “z***n”, performed abnormally. In the middle stage, he has a high bidding ratio (32% of bids), and he also outbidded himself twice: one time with four consecutive bids and another time with three. At the same time, his last bid of \$157.5 is less than the reserve price \$193.38. Our monitoring system will cluster him as “suspicious” according to his fraud score of 0.5424, he will receive a warning. Fortunately, this auction can be successfully finished.

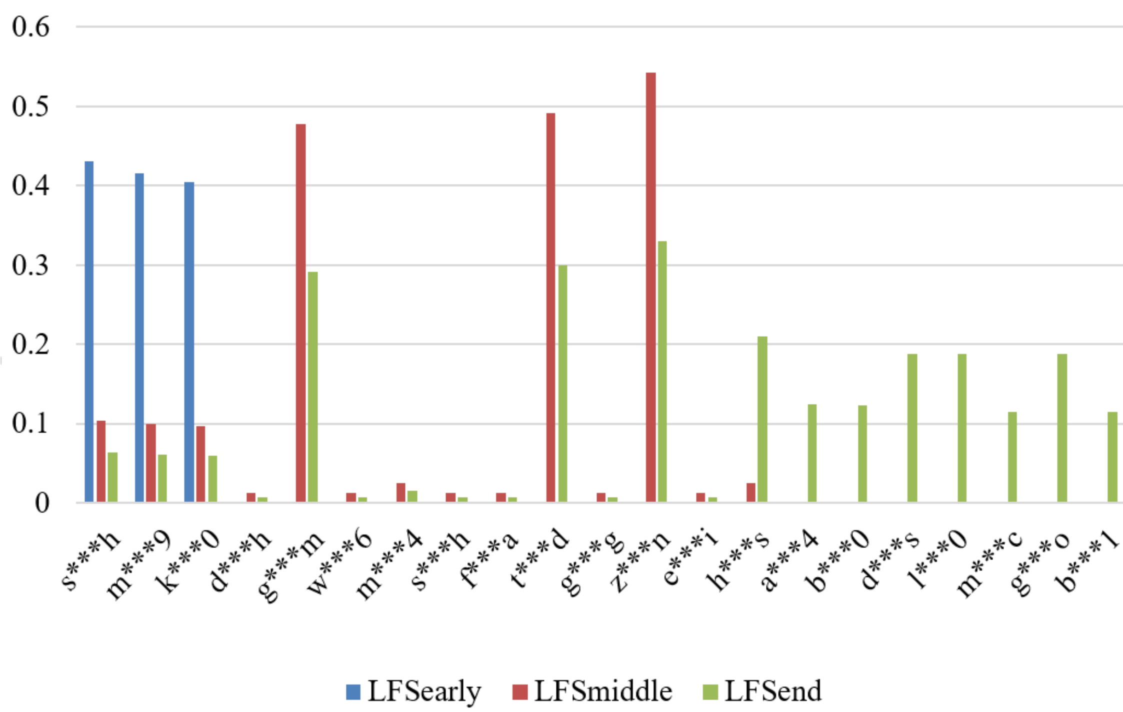

Figure 5.5: Runtime Detection in Auction #2.

5.3 Monitoring XBOX Auctions

In the XBOX auction dataset, the seller IDs are missing. However, it is needed to measure the buyer tendency metric. According to the real dataset of Palm PDA, the ratio of total auctions to total bidders is 149:71. As depicted in Figure 5.6, we generated 44 sellers to keep the same ratio. The figure shows the number of auctions hosted by each seller. Since shill bidding happens in auctions with more bids, we would like to monitor the auction with the highest number of bids (which is 75 bids). Table 5.7 exposes the statistical information of the largest Xbox auction initiated by seller "038". Similarly as the previous experiments, the live fraud scores of each participant as well as the bidding behaviour analysis of the shill bidders are given in Figure 5.7 and Table 5.8 respectively.

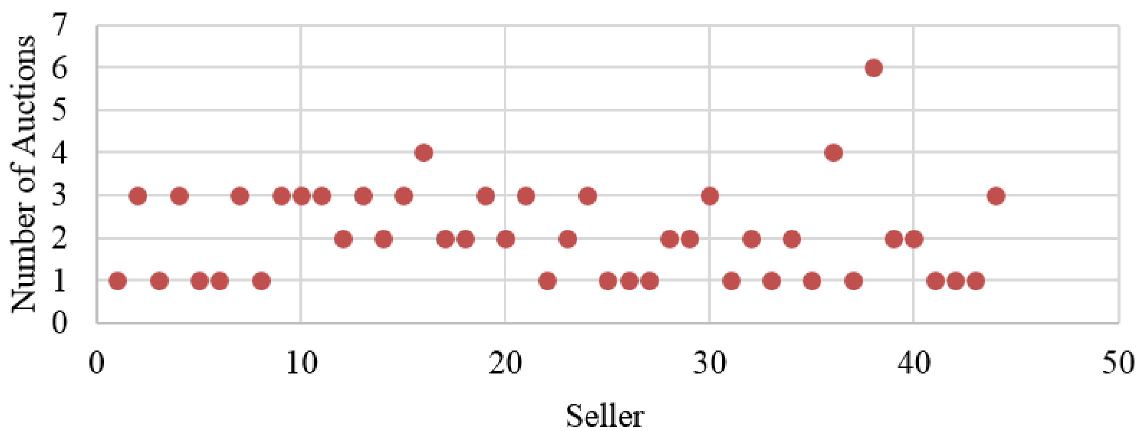

Figure 5.6: Sellers Distribution in XBOX Dataset.

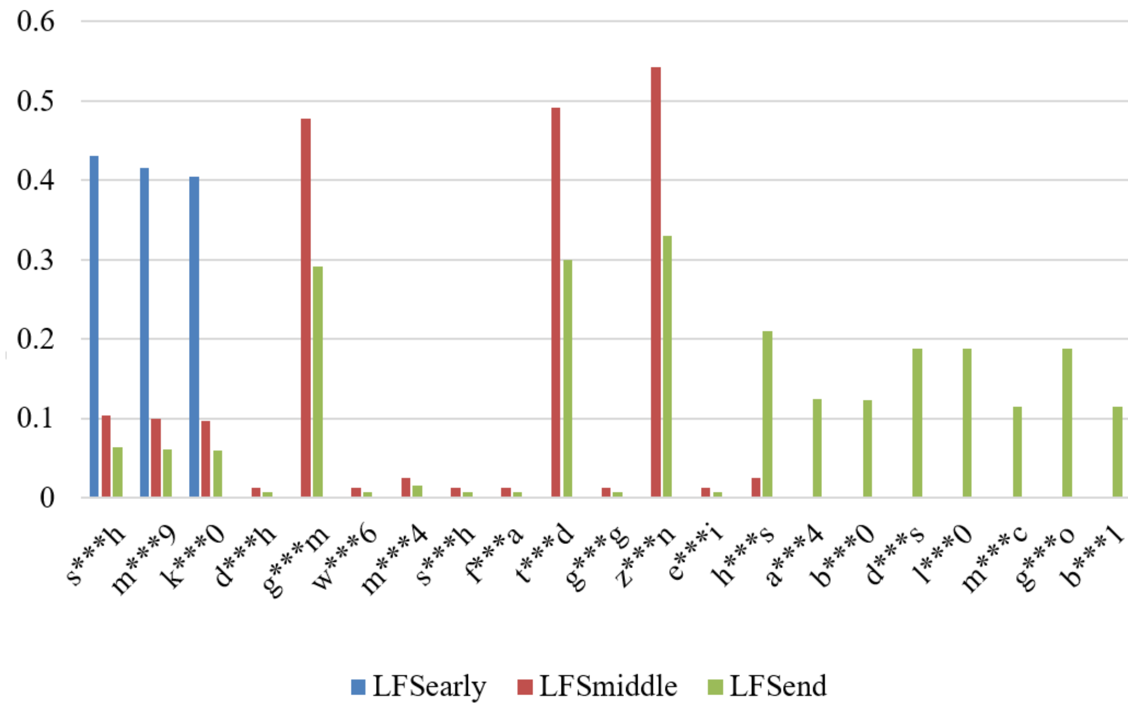

Figure 5.7: Runtime Detection in XBOX Auction.

Subsequently, the suspicious bidder “b***8” will be warned by our IAFM at the early stage while the fraudster “m***r” will be blamed at the middle stage. In fact, this Xbox auction should have terminated at the end of the middle stage. After the auction has been cancelled, the cluster of “b***8” will be updated to “suspicious” by considering his past and current fraud scores. The cluster of “m***r” will be labelled as “fraudulent” and therefore his account will be suspended.

5.4 Discussion

In our experiments, we found two auctions that were seriously infected with shill bidding. These two auctions should have been cancelled and the confirmed fraudster suspended. In the Palm PDA dataset, seller “s***l” held 26.8% auctions in the interval of 19 days. His auctions attracted 372 out of 1024 bidders. Among these 372 bidders, 84.1% bidders only participated in his auctions. And in one of his auction, only two bidders participated. Meanwhile, we found that his 2 top bids auctions were infected by shill bidding, in which one should be cancelled and one bidder in another should be warned. After examination, we noticed that 17.2% bidders who participated in his auctions are new, because the feedback rating is zero. Consequently, seller "s***l" is highly suspicious, and this information can be transmitted to eBay for further investigation.

bidderID	**Early Stage**
t***6	1. At only 7% of the elapsed auction time, he placed his first bid (amount is $5), which is the second bid in this auction **(EB)**. 2. According to his past bid activities, including the current auction, he participated 100% in the auctions held by seller “s***l” **(EB)**.
k***0	1. His first bidding time is at 17% of the auction duration **(EB)**, and with a bid amount of $38.88 which is much higher than the previous bid of $20. 2. Considering the bidder tendency, he participated 21 times in total and only for the seller “s***l” (100%) **(BT)**.
	Middle Stage
m***m	1. Around 47% of the total bids are placed by him in this stage **(BR)**. 2. His successive outbidding is 100% **(SO)**. And the difference between his first bid and last bid in this stage is $115. 3. He aggressively bided in this stage **(BR)** because the current price is less than the reserve price **(RPS)**.
	Final Stage
m***m	1. Since the current auction price is still $30 less than the reserve price, he continuously placed 6 bids within 1 hour. After the bid amount is $12 greater than the reserve price, he stopped bidding 10 hours before the auction ended **(LB)**. 2. This auction has a total of 44 bids, which is 2 times greater than the average number of bids **(AB)**. 3. He never won in any auction **(WR)**.

Table 5.4: Analysis of Suspicious/Fraudulent Bidders’ Behaviours.

$\mathbf{T_{early}}$		$\mathbf{T_{middle}}$		$\mathbf{T_{final}}$	
BidderID	Action	BidderID	Action	BidderID	Action
t***6	Blame	m***m	Warn	m***m	Warn
k***0	Blame				

Table 5.5: Runtime Reaction in Auction #1.

Palm Pilot M515 PDA (2013(4/14 4/20))-AuctionID: 3020532816			
Auction Details		Statistics (%)	
Starting price	$0.01	1-time bid	61.9%
Total bids	51	2-time bid	9.5%
Total bidders	21	Early stage	21.6%
Reserve price	$193.38	Middle stage	49%
Winning price	$227.5	Final stage	29.4%

Table 5.6: Auction #2 Statistical Information.

XBOX Game Console - AuctionID: 8214355679			
Auction Details		Statistics (%)	
Starting price	$0.99	1-time bid	27.3%
Total bids	75	2-time bid	18.2%
Total bidders	14	Early stage	28.0%
Reserve price	$208.25	Middle stage	26.7%
Winning price	$245	Final stage	45.3%

Table 5.7: Statistical Information of XBOX Dataset.

bidderID	**Early Stage**
b***8	1. He placed his first bid at 1% of the auction duration **(EB)**. 2. After checking his past history, 33.3% of participation activity is for the seller "038" **(BT)**.
	Middle Stage
m***r	1. Around 75% of bids are placed by this user in this stage, which is extremely high **(BR)**. 2. He aggressively placed bids, with a 100% successively outbidding: in one time 9 consecutive bids and in another time 5 ones **(SO)**. 3. His last bid is $105 which is still less than the reserve price of $208.25 **(RPS)**.

Table 5.8: Behaviour Analysis of Shill Bidders

Chapter 6

Conclusion and Future Work

This chapter is organized into two sections. Section 6.1 shows the conclusions of our approach and Section 6.2 presents the future work in this research area.

6.1 Conclusion

Research in fraud detection is becoming critical in order to establish trust in online applications and businesses. Online auctions are still not trustworthy due to the lack of runtime monitoring services. This lack allows auction users to fake their identities and behave as they desire. Without a rigorous monitoring system, online auctions will lead to a negative impact on innocent bidders.

In this thesis, our generic run-time monitoring system first detects In-Auction Fraud (IAF) by examining each bidder's activities based on the most reliable IAF patterns. After detecting abnormal behaviours in ongoing auctions, our system takes actions immediately by notifying the bidders at fault, cancelling the IAF-infected auctions, and/or suspending fraudsters' accounts. In this way, we increase the confidence of bidders for the online auctions. Each bidder has a stage live fraud score as well as an overall fraud score denoting his current misconduct in each live auction and in all the participated auctions respectively. Developing a stage-based run-time fraud monitoring service is substantially different than has been proposed in the very few studies on run-time IAF detection. We proposed an adaptive architecture for our monitoring system to be able to handle the scalability and run-time performance issues since hundreds of auctions operate concurrently in commercial auctions. Our system is designed with a dynamic architecture where several detection and reaction agents are added and removed during run-time. Our system can monitor a very large number of auctions. Every time auction is completed or terminated, the monitoring agent is deleted in order to release resources for the other agents. By using benchmark datasets, we monitored two Palm PDA auctions and one XBOX auction conducted in different years. The monitoring detected dishonest bidders in these three large auctions. Only one auction should have been successful. Indeed, one of the PDA auction should have been terminated in the early stage, and the XBOX auction in the middle stage.

6.2 Future Work

There are several interesting research directions of our study as described below:

Optimizing shill pattern weights: To search for the optimal weights of the shill patterns, we may utilize a machine learning method, such as the artificial networks as done in [3]. Other than the weights subjectively decided by the authors or experiments [8], using neural networks is more persuasive and reliable. The selected weights are combined linearly to obtain shill score. The optimal weights could come out by regulating every single weights until the best performance is found. Since it is based on systematically searching, the most optimal weights must be acquired.

Applying supervised machine learning techniques: We would like to apply a supervised machine learning technique, such as the robust Support Vector Machines (SVMs) [45] in order to monitor very large sets of auctions and users. The commercial auction data may be collected with a Web Scraper.

Employing a SVM-based multi-class classifier: Another good extension of our work, but still challenging, would be to employ a SVM-based multi-class classifier [46] to be able to classify bidders into three categories: “Normal”, “Suspicious”, and “Fraudulent”.

REFERENCES

[1] P. Unite, "Power sellers unite." URL: http://www.powersellersunite.com, September 2015.

[2] B. J. Ford, H. Xu, and I. Valova, "A real-time self-adaptive classifier for identifying suspicious bidders in online auctions," *The Computer Journal*, vol. 56, no. 5, pp. 646–663, 2013.

[3] S. Tsang, Y. S. Koh, G. Dobbie, and S. Alam, "Detecting online auction shilling frauds using supervised learning," *Expert Systems with Applications*, vol. 41, no. 6, pp. 3027–3040, 2014.

[4] W.-H. Chang and J.-S. Chang, "A novel two-stage phased modeling framework for early fraud detection in online auctions," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11244–11260, 2011.

[5] IC3, "The internet crime complaint center (ic3)." URL: http://www.ic3.gov, September 2015.

[6] F. Dong, S. M. Shatz, and H. Xu, "Combating online in-auction fraud: Clues, techniques and challenges," *Computer Science Review*, vol. 3, no. 4, pp. 245–258, 2009.

[7] J. Trevathan and W. Read, "Undesirable and fraudulent behaviour in online auctions.," in *the International Conference on Security and Cryptography Conference (SECRYPT)*, pp. 450–458, Citeseer, 2006.

[8] J. Trevathan and W. Read, "Detecting shill bidding in online english auctions," *Handbook of research on social and organizational liabilities in information security*, pp. 446–470, 2009.

[9] H. Xu, S. M. Shatz, and C. K. Bates, "A framework for agent-based trust management in online auctions," in *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*, pp. 149–155, IEEE, 2008.

[10] F. Dong, S. M. Shatz, and H. Xu, "Reasoning under uncertainty for shill detection in online auctions using dempster–shafer theory," *International Journal of Software Engineering and Knowledge Engineering*, vol. 20, no. 07, pp. 943–973, 2010.

[11] H. Xu, C. K. Bates, and S. M. Shatz, "Real-time model checking for shill detection in live online auctions.," in *Software Engineering Research and Practice*, pp. 134–140, 2009.

[12] S. Sadaoui, X. Wang, and D. Qi, "A real-time monitoring framework for online auctions frauds," in *Current Approaches in Applied Artificial Intelligence*, pp. 97–108, Springer, 2015.

[13] J. Trevathan and W. Read, "Investigating shill bidding behaviour involving colluding bidders," *Journal of Computers*, vol. 2, no. 10, pp. 63–75, 2007.

[14] R. J. Kauffman and C. A. Wood, "Irregular bidding from opportunism: An exploration of shilling in online auctions," *Information Systems Research, VV*, pp. 1–36, 2007.

[15] J. Engelberg and J. Williams, "ebays proxy bidding: A license to shill," *Journal of Economic Behavior Organization*, vol. 72, no. 1, pp. 509–526, 2009.

[16] A. Goel, H. Xu, and S. M. Shatz, "A multi-state bayesian network for shill verification in online auctions," in *Software Engineering and Knowledge Engineering (SEKE)*, pp. 279–285, 2010.

[17] T. Yoshida and H. Ohwada, "Shill bidder detection for online auctions," in *the Pacific Rim International Conference on Artificial Intelligence (PRICAI): Trends in Artificial Intelligence*, pp. 351–358, Springer, 2010.

[18] A. Mundra and N. Rakesh, "Online hybrid model for online fraud prevention and detection," in *Intelligent Computing, Networking, and Informatics*, pp. 805–815, Springer, 2014.

[19] F. Dong, S. M. Shatz, H. Xu, and D. Majumdar, "Price comparison: A reliable approach to identifying shill bidding in online auctions?," *Electronic Commerce Research and Applications*, vol. 11, no. 2, pp. 171–179, 2012.

[20] D. H. Chau, S. Pandit, and C. Faloutsos, "Detecting fraudulent personalities in networks of online auctioneers," in *Knowledge Discovery in Databases: PKDD 2006*, pp. 103–114, Springer, 2006.

[21] B. Lei, H. Zhang, H. Chen, L. Liu, and D. Wang, "A k-means clustering based algorithm for shill bidding recognition in online auction," in *Control and Decision Conference (CCDC), 2012 24th Chinese*, pp. 939–943, IEEE, 2012.

[22] B. J. Ford, H. Xu, and I. Valova, "Identifying suspicious bidders utilizing hierarchical clustering and decision trees," in *IC-AI*, pp. 195–201, 2010.

[23] P. Bangcharoensap, H. Kobayashi, N. Shimizu, S. Yamauchi, and T. Murata, "Two step graph-based semi-supervised learning for online auction fraud detec-

tion," in *Machine Learning and Knowledge Discovery in Databases*, pp. 165–179, Springer, 2015.

[24] C.-H. Yu and S.-J. Lin, "Fuzzy rule optimization for online auction frauds detection based on genetic algorithm," *Electronic Commerce Research*, vol. 13, no. 2, pp. 169–182, 2013.

[25] M. D.S.L., S. M., and N. M.V.S.S., "Machine learning approach to handle fraud bids," *International Journal for Development of Computer Science and Technology*, vol. 1, no. 5, pp. 56–61, 2013.

[26] C.-H. Yu and S.-J. Lin, "Parallel crawling and capturing for on-line auction," in *Intelligence and Security Informatics*, pp. 455–466, Springer, 2008.

[27] R. Balingit, J. Trevathan, Y. J. Lee, and W. Read, "A software tool for collecting data from online auctions," in *Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on*, pp. 922–927, IEEE, 2009.

[28] 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 16th international conference on World Wide Web*, pp. 201–210, ACM, 2007.

[29] L. Khomnotai and J.-L. Lin, "Detecting fraudsters in online auction using variations of neighbor diversity," *International Journal of Engineering and Technology*, vol. 5, no. 3, pp. 156–164, 2015.

[30] S. Tsang, G. Dobbie, and Y. S. Koh, "Evaluating fraud detection algorithms using an auction data generator," in *Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on*, pp. 332–339, IEEE, 2012.

[31] P. Ravisankar, V. Ravi, G. R. Rao, and I. Bose, "Detection of financial statement fraud and feature selection using data mining techniques," *Decision Support Systems*, vol. 50, no. 2, pp. 491–500, 2011.

[32] K. P. Sycara, "Multiagent systems," *AI magazine*, vol. 19, no. 2, pp. 79–92, 1988.

[33] M. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practice," *The knowledge engineering review*, vol. 10, no. 02, pp. 115–152, 1995.

[34] Y. Wang, J. Zhang, and J. Vassileva, "A super-agent-based framework for reputation management and community formation in decentralized systems," *Computational Intelligence*, vol. 30, no. 4, pp. 722–751, 2014.

[35] O. Shehory and A. Sturm, "Multi-agent systems: A software architecture viewpoint," in *Agent-Oriented Software Engineering*, pp. 57–78, Springer, 2014.

[36] H. C. Wong and K. Sycara, "A taxonomy of middle-agents for the internet," in *the International Conference on Computer Communication and Networks (IC-CCN)*, vol. 1757, pp. 456–466, IEEE, 2000.

[37] O. Shehory, "A scalable agent location mechanism," in *Intelligent Agents VI. Agent Theories, Architectures, and Languages*, pp. 162–172, Springer, 2000.

[38] D. Ben-Ami and O. Shehory, "A comparative evaluation of agent location mechanisms in large scale mas," in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pp. 339–346, ACM, 2005.

[39] M. Bratman, "Intention, plans, and practical reason," 1987.

[40] A. Rao, S. Georgeff, and M. P, "Bdi agents: From theory to practice.," in *International Conference on MultiAgent Systems (ICMAS)*, vol. 95, pp. 312–319, 1995.

[41] L. Braubach, A. Pokahr, and W. Lamersdorf, "Jadex: A bdi-agent system combining middleware and reasoning," in *Software agent-based applications, platforms and development kits*, pp. 143–168, Springer, 2005.

[42] A. Pokahr, L. Braubach, C. Haubeck, and J. Ladiges, "Programming bdi agents with pure java," in *Multiagent System Technologies*, pp. 216–233, Springer, 2014.

[43] D. Lucking-Reiley, D. Bryan, N. Prasad, and D. Reeves, "Pennies from ebay: The determinants of price in online auctions*," *The Journal of Industrial Economics*, vol. 55, no. 2, pp. 223–233, 2007.

[44] G. Kołaczek and S. Balcerzak, "Identification of shill bidding for online auctions using anomaly detection," in *New Trends in Intelligent Information and Database Systems*, pp. 111–120, Springer, 2015.

[45] S. Zhang, S. Sadaoui, and M. Mouhoub, "An empirical analysis of imbalanced data classification," *Computer and Information Science*, vol. 8, no. 1, pp. 151–159, 2015.

[46] Z.-X. Yang, Y.-H. Shao, and X.-S. Zhang, "Multiple birth support vector machine for multi-class classification," *Neural Computing and Applications*, vol. 22, no. 1, pp. 153–161, 2013.

Appendix A

Implementation of Agents

In this Chapter, we present the Java source code of BDI agents: AucController agent, AucMonitoring agent and ClusterUpdating agent.

A.1 AucControllerBDI Agent

```
package a1;
import jadex.bdiv3.BDIAgent;
import jadex.bdiv3.annotation.Body;
import jadex.bdiv3.annotation.Plan;
import jadex.bdiv3.annotation.Plans;
import jadex.micro.annotation.Agent;
import jadex.micro.annotation.AgentBody;
import jadex.micro.annotation.Description;

@Agent
@Description("<h1>Auction␣Controller␣Agent</h1><br>"
                + "<b>Functionalities:</b><br>"
                + "1.␣Creating␣Monitoring␣Agents<br>"
                + "2.␣Notifying␣suspicious␣participants<br>"
                + "3.␣Extracting␣Raw␣Auction␣Data<br>")
```

```
@Plans({
        @Plan(body=@Body(CreateAucMonitoringPlan.class)),
        @Plan(body=@Body(ExtractRawDataPlan.class))
})
public class AucControllerBDI {
        @Agent
        public BDIAgent aucControllerAgent;

        @AgentBody
        public void body(){

                // Controller agent adopt the plan of creating
                   ↪ AucMonitoringAgent
                CreateAucMonitoringPlan createAucMonitoringPlan
                        = new CreateAucMonitoringPlan();
                aucControllerAgent.adoptPlan(createAucMonitoringPlan
                   ↪ );

                // Get auction data from database
                // 1. Call the Plan: ExtractRawDataPlan
                // 2. Passing to Monitoring Agent
                ExtractRawDataPlan extractRawDataPlan
                        = new ExtractRawDataPlan();
                aucControllerAgent.adoptPlan(extractRawDataPlan);

                // Notify Bidders
                NotifyPartsPlan notifyPartsPlan = new
                   ↪ NotifyPartsPlan();
                aucControllerAgent.adoptPlan(notifyPartsPlan);
        }
}
```

A.2 AucMonitoringBDI Agent

```
package a1;
import java.sql.SQLException;
import java.util.ArrayList;
import javax.swing.JFrame;
import jadex.bdiv3.BDIAgent;
import jadex.bdiv3.annotation.Body;
import jadex.bdiv3.annotation.Plan;
import jadex.bdiv3.annotation.Plans;
import jadex.bridge.service.RequiredServiceInfo;
import jadex.bridge.service.search.SServiceProvider;
import jadex.commons.future.IntermediateDefaultResultListener;
import jadex.micro.annotation.Agent;
import jadex.micro.annotation.AgentBody;
import jadex.micro.annotation.Description;


@Agent
@Description("<h1>A Dynamic Stage-based Fraud Monitoring System</h1
    ↪ >" + "<b>Functionalities:</b><br>"
                + "1. Triggering Cluster Updating Agent<br>" + "2.
                   ↪ Self-Destorying Agent<br>"
                + "3. Computing Live Fraud Score<br>" + "4.
                   ↪ Reporting Detection Results<br>"
                + "5. Extracting Past and Real Data<br>")
@Plans({ @Plan(body = @Body(GetRealPastDataPlan.class) ), @Plan(body
    ↪  = @Body(SelfDestoryPlan.class) ),
                @Plan(body = @Body(ReportDetectionPlan.class) ),
                @Plan(body=@Body(NotifyPartsPlan.class))})
public class AucMonitoringBDI extends JFrame {
        private static final long serialVersionUID = 1L;
        @Agent
        protected BDIAgent aucMonitoringAgent;
```

```
@AgentBody
public void body() throws SQLException {
        // Get the authorization to work
        SServiceProvider .getServices(aucMonitoringAgent.
           ↪ getServiceProvider(),
           ↪ ICreateAucMonitoringService.class,
           ↪ RequiredServiceInfo.SCOPE_PLATFORM).
           ↪ addResultListener(new
           ↪ IntermediateDefaultResultListener<
           ↪ ICreateAucMonitoringService>() {public void
           ↪ intermediateResultAvailable(
           ↪ ICreateAucMonitoringService ts) {
if (ts.createMonitoringAgent().get() == 1) {
        System.out.println("Monitoring Agent Created
           ↪ Successfully");
} else {
        System.out.println("Please check for re-creating");
}
}});
        System.out.println("Monitoring Agent Created
           ↪ Successfully");
        ArrayList<Double> patternScore = new ArrayList<
           ↪ Double>();
        GetRealPastDataPlan getRealPastDataPlan = new
           ↪ GetRealPastDataPlan();
        // aucMonitoringAgent.adoptPlan(getRealPastDataPlan)
           ↪ ;
        for (int i = 1; i <= 14; i++) {
                patternScore = getRealPastDataPlan.getData(i
                   ↪ );
                ComputeLFSPlan.computPatternScore(
                   ↪ patternScore, i);
        }
```

```
                // Report the detection result to Controller Agent
                aucMonitoringAgent.adoptPlan(new ReportDetectionPlan
                    ↪ ());

                // Notify Parts
                aucMonitoringAgent.adoptPlan(new NotifyPartsPlan());
                // Self Destroy Plan
                // After Processing the Live Fraud Score, Monitoring
                    ↪  Agent kill himself;
                aucMonitoringAgent.adoptPlan(new SelfDestoryPlan());
        }
}
```

A.3 ClusterUpdating Agent

```
package a1;
import jadex.bdiv3.BDIAgent;
import jadex.bdiv3.annotation.Body;
import jadex.bdiv3.annotation.Plan;
import jadex.bdiv3.annotation.Plans;
import jadex.micro.annotation.Agent;
import jadex.micro.annotation.AgentBody;
import jadex.micro.annotation.Description;
/*
 * This is the cluster updating agent
 * 1. after the monitoring is done, it will be trigger to upate
 * the user profile.
 * 2. update the auction log.
 *
 */
@Agent
@Description("<h1>Cluster Updating Agent</h1>"
                + "<b>Functionalities:</b><br>"
                + "1. Updating User Table <br>"
                + "2. Updating Auction Table <br>")
@Plans({
        @Plan(body = @Body(UpdateUserTablePlan.class) ),
        @Plan(body=@Body(UpdateAucTablePlan.class))
})
public class ClusterUpdatingBDI {
        @Agent
        public BDIAgent clusterUpdateAgent;
        @AgentBody
        public void body(){
                // adopt plan for updating user cluster
```

```
                    clusterUpdateAgent.adoptPlan(new UpdateUserTablePlan
                        ↪ ());
          }
}
```