

# Real-Time Model Checking for Shill Detection in Live Online Auctions\*

Haiping Xu<sup>1</sup>, Christopher K. Bates<sup>1</sup>, and Sol M. Shatz<sup>2</sup>

<sup>1</sup>Computer and Information Science Department

University of Massachusetts Dartmouth, North Dartmouth, MA 02747, USA

E-mail: {h xu, u\_cbates}@umassd.edu

<sup>2</sup>Computer Science Department

University of Illinois at Chicago, Chicago, IL 60607, USA

E-mail: shatz@uic.edu

**Abstract** - Online auctions are vulnerable to shill bidders, who intend to artificially raise bidding prices, causing winning bidders to pay more than they should pay for auctioned items. Detection of such fraudulent behaviors is very difficult, especially when an auction is in progress, or “live”. This paper focuses on a formal technique to detect shilling behaviors in live online auctions. We define a monitoring agent that can continuously watch for abnormal bidding behaviors of a monitored bidder. To make the detection process efficient, we introduce a dynamic auction model (DAM), and use real-time model checking techniques to verify shilling behaviors specified in linear temporal logic (LTL). Finally, we present an algorithm for real-time shill detection, and use a case study to demonstrate the efficiency and effectiveness of our approach.

**Keywords:** Online auctions, model checking, shilling behaviors, shill detection, real-time, software agents.

## 1 Introduction

Auctions have been a popular mechanism for purchasing items for a long time, providing potential buyers with a means to purchase an item at a price lower than normal cost or to purchase a rare item not otherwise available. As auction activities around the world have begun to spread, it becomes more and more difficult to reach intended audiences as they could be far away from the places where the auctions are conducted. The Internet has provided a solution to this, giving access to an auction to anyone anywhere in the world. This has opened the auction environment to the e-commerce world. However, since anyone can participate in an auction using a fake identification, it becomes very difficult for an auction house to determine who is actually participating in an auction. Thus, security concerns become an important

issue, and fraudulent bidders, typically known as *shill bidders*, may damage the validity of an auction by raising the price higher than normal, in an attempt to earn the seller a higher profit. Detection of shill bidders can be very difficult as most shilling behaviors resemble normal bidding behaviors to avoid being detected. Previous attempts for shill detection include identifying patterns for shilling behaviors and searching for matched shilling behaviors by investigating auction histories [1, 2]. By studying bidding behaviors, it is possible to find shill patterns, and therefore to define corresponding shill detection rules and mechanism. However, previous efforts only focus on identifying shill bidders using historical auction data. Such approaches may be useful for identifying shill patterns, but they are not very effective in shill detection since any detection happens after the auction is completed. Unfortunately, a completed auction that involved shill bidding has already resulted in possible losses for bidders, and also damaged the reputation of the auction house. In contrast, in this paper, we introduce a formal approach to detecting shilling behaviors in auctions that are in progress, which we refer to as “live auctions.” Using our approach, we can take appropriate actions timely on shilling behaviors, e.g., giving a shill bidder a warning or canceling the involved live online auction.

Our method is based on a model checking approach we proposed previously [2]. In our previous efforts, we demonstrated how to use model checking techniques to detect shill bidders using collected offline auction data, particularly from concurrent online auctions. In this paper, we extend our previous model by introducing a dynamic auction model (DAM) for real-time detection of shilling behaviors. In our approach, we first divide the duration of an online auction into three stages (i.e., *early*, *middle* and *final stage*), and then formally specify shilling behaviors in different stages in linear temporal logic (LTL). Finally, we run the SPIN model checker [3] on DAM in order to identify abnormal bidding behaviors. Note that DAM can be used to simulate an online auction using real-time auction data. As more and more shilling behaviors are identified for a monitored bidder, shilling scores for that

---

\* This material is based upon work supported by the U.S. National Science Foundation under grant numbers CNS-0715648 and CNS-0715657.

bidder can be accumulated. When the total shilling score reaches certain limits, appropriate actions can be taken against the monitored bidder. For example, a high shilling score may result in a warning of suspected shilling behaviors, while a very high shilling score may result in cancellation of the involved online auction.

Most previous work on shill detection in online auctions is based on analyzing large volumes of historical auction data to search for shill patterns. Kauffman and Wood used a statistical approach to detecting shilling behaviors and showed how the statistical data of a market would look if opportunistic behaviors do exist [4]. They also showed how to use an empirical model to test for questionable behaviors. However, one limitation of the approach is the need to review multiple auctions over a long period of time [5]. Furthermore, since the statistical approach was based on analyzing a large amount of historical auction data, it was not applicable to directly analyzing a particular auction where shilling behaviors might be involved. Similarly, Chau, et al. from Carnegie Mellon University proposed a shill detection method, called 2-Level Fraud Spotting (2LFS), which can be used to detect fraudsters in online auctions using data mining techniques by investigating historical auction data from eBay [1]. Other related work on shilling behaviors includes attempts to get around the shilling problem by designing sound mechanisms to decrease the incentives for shilling behaviors. For example, researchers have proposed reputation mechanisms in online auctions to deter opportunistic behaviors [6, 7]. However, acquainted users may put in good comments for each other, and thus, the reputation system can be easily manipulated. An improved reputation-based approach is to develop models that characterize sellers according to statistical metrics related to price inflation [8]. Unfortunately, such an approach is also not suitable to detect shill bidders in real-time because it is based on analyzing large volumes of auction data.

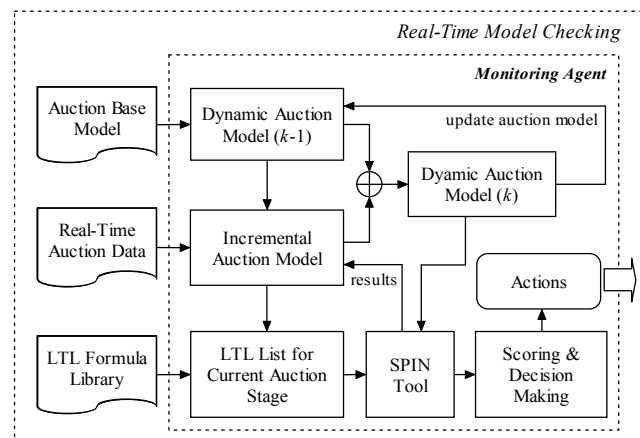
Unlike the above approaches, our approach uses real-time auction data, so abnormal bidding behaviors can be captured timely. Our approach complements existing approaches, such as the statistical approach, for shill detection. Although statistical approaches require analysis of large volumes of historical auction data, they can effectively identify bidding patterns, which can be adopted as useful knowledge in our model checking approach.

Other related work includes trustworthy management in e-commerce for securing online transactions and establishing trust in users by proposing different types of trust models. Trust management using reputation models are based on prior history of users and/or feedback gathered from other entities. Shmatikov and Talcott proposed a formal model that precisely defined the notion of reputation, and can be used to reason about trust [9]. Boukerche and Xu proposed an agent-based trust and reputation local storage strategy for wireless sensor networks, which can manage trust and reputation with minimal overhead in terms of extra messages and time delay [10]. Recently, we introduced a framework for

agent-based trust management (ATM) in online auctions [11]. In this paper, we enhance the ATM framework by providing a real-time model checking mechanism for monitoring agents, which can continuously watch for abnormal bidding behaviors and take actions accordingly. Since our detection mechanism only deals with real-time auction data, a monitoring agent can efficiently and effectively report shilling behaviors.

## 2 Agent-based shill detection

Agent technology provides a programming paradigm for development of intelligent software with the capability of sensing, planning, scheduling, reasoning and decision-making. In order to automatically detect shilling behaviors in real-time and take appropriate actions autonomously, we define a monitoring agent for each bidder who registers with an auction house. Each monitoring agent is built with enough capability to work independently, and can keep track of live auctions that the bidder is involved with. As an auction progresses, a dynamic auction model for the monitored bidder is continuously updated based on real-time auction data, and verified by the monitoring agent against formal specifications of shilling behaviors. Figure 1 presents an overview of a monitoring agent using real-time model checking.



**Figure 1.** Monitoring agent using real-time model checking

From the figure, we can see that the dynamic auction model is initially created using an auction base model, which models the basic functionality of an online auction. The dynamic auction model is then updated each time the monitoring agent starts to perform verifications. The new dynamic auction model (for iteration  $k$ ) is a combination of the previous dynamic auction model (for iteration  $k-1$ ) and the incremental auction model created based on real-time auction data. A list of LTL formulas is then selected from an LTL formula library for verification of shilling behaviors. Finally, the monitoring agent uses the SPIN model checker to compute the verification results, and also makes decisions on taking appropriate actions.

When the above process is complete, the dynamic auction model ( $k-1$ ) is updated for the next iteration using

the dynamic auction model ( $k$ ). The model checking results are also recorded for use in updating the incremental auction model in the next iteration. Thus, all previous verification results can be directly used for further verification in later iterations.

Note that in order to select suitable LTL formulas for formal verification, we divide an auction duration into three stages, namely *early stage*, *middle stage* and *final stage*, which are defined as follows.

**Definition 2.1 Early Stage.** The early stage of an online auction is defined as the first quarter of the auction duration. Typically there are only a few bids placed, but a shill bidder may be eager to drive up the auction price as early as possible.

**Definition 2.2 Middle Stage.** Most of the online auction bidding activities occur at the middle stage of the auction duration. The middle stage is defined as  $[0.25T, 0.9T]$ , where  $T$  is the duration of the auction. Most of the shilling behaviors are expected to be detected in this stage.

**Definition 2.3 Final Stage.** The final stage of an online auction is defined as the last 10% of the auction time. In this stage, a shill only places bids occasionally and very carefully in order to avoid winning the auction.

To make our approach efficient, the dynamic auction model only simulates the bidding activities for the current auction stage, because all shilling behaviors detected in previous stages have already been recorded in the model.

### 3 Real-time model checking for shill detection

#### 3.1 Dynamic auction model

The dynamic auction model (DAM) records bids and uses variables or flags to represent auction states. It also simulates a live online auction in real-time, as specified in PROMELA (Process Meta Language), which is a formal language that allows for the dynamic creation of concurrent processes [3]. DAM consists of two major components, namely the current dynamic auction model (CDAM), and the incremental auction model (IAM). The initial CDAM is instantiated using a base auction model, which provides the primary variable declarations as well as the code that simulates an online auction in real-time. Figure 2 illustrates the sample PROMELA code for an example initial current dynamic auction model.

From the sample code, we can see that the declared variables are either related to the auction itself or to the monitored bidder. The model first defines an auction *auc* of type *Auction* that consists of variables such as *startTime*, *endTime*, *estimatedPrice*, and *reservePrice*. Then a list of bids *bids* is defined as an array of bids of type *Bid*, where we assume that the maximum number of bids in a single auction is 100. Additional system variables are defined, for example variable *numberOfBids* refers to the number of bids that have already been placed in the auction at the

model checking time, and variable *startingIndex* is the index of the first bid for the current auction stage. Furthermore, in order to verify shilling behaviors of the monitored bidder, we define a set of variables for the monitored bidder, such as *monitoredBid* and *monitoredInc*, which are used when defining temporal formulas.

```

/*type and variable declaration*/
typedef Auction {
  int startTime = 0;
  int endTime = 172800;
  short estimatedPrice = 1500;
  short reservePrice = 1350;
  short minIncrement = 5;
} auc;
typedef Bid{
  short bidderID; // bidder's identification
  short bidAmount; // bid amount in dollars
  int bidTime; // time when bid is placed
} bids[100];
short numberOfBids; // number of bids so far
short startingIndex; // for current stage
int middleStageStart; // middle stage start time
int finalStageStart; // final stage start time
...
short monitoredBidderID = 000001;
short monitoredBid; // bid amount in dollars
short monitoredInc // bid increment
bit bidFlag; // set to 1 if the current
// bid is monitored
...
/* placeholder for previous auction data */
...
/* previous model checking results */
...
proctype SimulateBiddingProcess() {
  int index = startingIndex;
  ...
  do
  ::(index < numberOfBids) ->
    d_step {
      bidFlag = 0; // reset bid status
      ...
      if
      ::(index > 0) ->
        previousBid=bids[index-1].bidAmount;
      ::(index == 0) -> previousBid = 0;
      fi;
      if /* bid is monitored */
      ::(bids[index].bidderID ==
        monitoredBidderID)-> monitoredIncrement
        =bids[index].bidAmount-previousBid;
        monitoredBid=bids[index].bidAmount;
        bidFlag = 1;
      fi;
      ...
      index++;
    }
  od;
}

```

**Figure 2.** Sample PROMELA code for an initial CDAM

The real-time simulation of a live online auction is a simple replay of the auction activities occurred during the current auction stage. The simulation process is modeled by the *SimulateBiddingProcess* procedure, which is

activated whenever a monitored bidder places a bid or when there is an auction transition (e.g., a transition from the early stage to the middle stage). The major task of this procedure is to set values for certain variables that are used by the model checker. For example, the flag *bidFlag* indicates if the current bid is placed by a monitored bidder, and *monitoredInc* stores the bid increment for the current monitored bid. Note that the real-time auction data and new model checking results are initially recorded in IAM, which will be stored in CDAM as previous auction data and previous model checking results each time the two models are combined.

Figure 3 shows the sample PROMELA code for IAM, which represents the dynamic portion of a DAM.

```

typedef ShillingBehavior {
  bit detected;
  int timeDetected;
  int detectionCount;
}
/* a list of shilling behaviors to be checked
in the current auction stage */
ShillingBehavior BM1;
ShillingBehavior BM2;
ShillingBehavior BM3;
...
proctype CreateIncrementModel(){
  /* real time auction data */
  bids[26].bidID = 000003;
  bids[26].bidAmt = 885;
  bids[26].bidTime = 50424;
  bids[27].bidID = 000004;
  bids[27].bidAmt = 900;
  bids[27].bidTime = 50580;
  ...
  numberOfBids = 30; // number of bids so far
  startingIndex = 21; // for middle stage
  ...
  /* update previous model checking results */
  BM1.detected = 1;
  BM1.timeDetected = 48450;
  BM1.detectionCount = 1;
  BM2.detected = 0;
  BM2.timeDetected = 0;
  BM2.detectionCount = 0;
  ...
}

```

Figure 3. Sample PROMELA code for IAM

As shown in Figure 3, IAM first defines the *ShillingBehavior* type, and then defines a set of shilling behaviors to be checked during the current auction stage, such as *BM1* and *BM2* (shilling behaviors in the middle auction stage, defined in Section 3.2). The major component of an IAM is the new real-time data collected from a live online auction as well as updated values for certain system variables, such as *numberOfBids* and *startingIndex*. Furthermore, the model checking results must be updated. For example, as shown in Figure 3, shilling behavior *BM1* has been detected once previously at time 48450. Such information is critical in calculating shilling scores for the monitored bidder.

### 3.2 Real-time model checking

When the SPIN model checker is initiated, LTL formulas representing shilling behaviors are incorporated into DAM. Each LTL formula to be checked is converted into a PROMELA *never* claim, which is appended to the end of the auction model [3, 4]. When the model checker is running, the *never* claim is verified against the auction model and returns whether the LTL formula is *valid* or *invalid*. If a formula is *valid*, it indicates the corresponding shilling behavior is detected; otherwise, no such shilling behavior is detected.

The verification of the model is actually done by generating C code of the model through SPIN. Once the source code is generated, it is compiled into an executable verifier, which is then run by the monitoring agent to determine the validity of a specified behavior. This process is repeated as long as the auction is active. Each time a shilling behavior is detected, the monitored bidder's shilling score is updated. When the score reaches a certain limit, an appropriate action must be taken.

The scoring of different shilling behaviors is used to indicate how malicious a bidder is. Some behaviors that match as shilling behaviors are not severe because it is possible that the bidder may have no malicious intent at all. In this case, a low score is assigned when the behavior is detected. Some other shilling behaviors may appear at an earlier stage, and prove to be benign later in the auction. In this case, a temporary score is assigned, which may be discarded later if needed.

As more shilling behaviors are identified during an online auction, a bidder's shilling score will increase. As the score increases, more harsh actions can be taken for the bidder. For example, a warning can be issued to the monitored bidder if the bidder shows some malicious behaviors but the shilling score is still not sufficiently high. On the other hand, a higher score may indicate the involved auction has been significantly affected; thus, the auction must be cancelled in order to protect the interests of other bidders who have been involved in the auction.

We summarize various shilling behaviors into three groups according to the three stages of an online auction, namely early stage, middle stage, and final stage. Tables 1-3 show a few examples of shilling behaviors in each auction stage.

Table 1. Examples of shilling behaviors in early stage

BID*	Shilling Behavior (Early Stage)
BE1	Bidding time very close to the start of an auction. <u>Explanation</u> : If a bid is placed on an auction very early, say within 4 hours after the start of the auction, then it is likely that the bidder has prior knowledge about the auction, and therefore is a likely shill.
BE2	Bid close to the reserve price with no larger bids in the early stage. <u>Explanation</u> : A normal early stage bid would not likely jump to a value that is close to the reserve price. This behavior casts strong suspicion of intent to stimulate the auction and encourage more bidding.

BE3	Large number of bids compared to other bidders in the early stage. <u>Explanation:</u> A skill bidder may try to raise the price as much as possible since there is little risk in the early stage. This results in a large portion of the bids on the auction to be made by the skill bidder to outbid others.
-----	--

\*BID: Behavior Identification

Table 2. Examples of shilling behaviors in middle stage

BID	Shilling Behavior (Middle Stage)
BM1	Bid close to the reserve price with no larger bids over the reserve price in the middle stage. <u>Explanation:</u> If the bidding price is not high enough, a shill may attempt to push the price higher with a bid close to the reserve price.
BM2	BE2 detected, and no bids greater than the reserve price in the middle stage. <u>Explanation:</u> When the reserve price is reached, the shill bidder stops bidding in order to avoid winning the auction.
BM3	BE2 detected, and bids with small bid increments over reserve price in the middle stage. <u>Explanation:</u> When the reserve price is reached, a shill bidder may try to drive the price up a little more with small increments to avoid winning the auction.
BM4	A much larger bid than the estimated price in the middle stage. <u>Explanation:</u> A bid that is 20% over the estimated price during the middle stage is not likely made by a normal bidder with a normal increment amount.

Table 3. Examples of shilling behaviors in final stage

BID	Shilling Behavior (Final Stage)
BF1	BM1 or BM2 detected, and no bids greater than the reserve price in the final stage. <u>Explanation:</u> Similar to BM2, except that BF1 appears in the final stage.
BF2	BM1 detected, and bids with small bid increments over the reserve price in the final stage. <u>Explanation:</u> Similar to BM3, except that BF2 appears in the final stage.

We now provide some examples of LTL formulas to show how shilling behaviors can be formally specified.

#### BE1: Bidding time very close to the start of an auction

Shill bidders tend to place bids more at the beginning of an auction than the end [12], giving more time for other bidders to outbid the shill bids. While normal bidders may place bids on an auction close to the auction's creation time, closer bids are a good indicator of prior knowledge about the auction. It is unlikely that a normal bidder would notice an auction right after the auction's creation, while shill bidders may place bids on their own auctions very early to attract potential legitimate bidders.

*Detection:* To detect this behavior, we use an LTL formula that represents a bid placed on an auction before the elapsed time of the auction has reached a certain time, say

4 hours. The formula defined below is modeled using *existence* LTL pattern "P becomes true before Q" [13].

```
#define p (bidFlag == 1)
#define q (elapsedAuctionTime > 14400)
(!!q || (!q U (p && !q)))
```

In this pattern, the formula holds true if event  $p$  (i.e.,  $\text{bidFlag} == 1$ ) becomes true before event  $q$  ( $\text{elapsedAuctionTime}$  is greater than 4 hours) occurs.

*Analysis:* Once shilling behavior  $BE1$  is detected, a quick analysis of the history can be done to determine how early the bid was placed. Based on the time that the bids were first placed on the auction, a shilling score is calculated.

#### BM1: Bid close to the reserve price with no larger bids over the reserve price in the middle stage

A reserve price is typically less than the estimated price of an auctioned item, which is defined in this paper as 90 percent of the estimated price. If the current bidding price has not yet been raised to the reserve price, a shill may attempt to push the price higher with bids close to the reserve price. However, placing bids higher than the reserve price could be risky for the shill bidder because the shill bidder may accidentally win the auction.

*Detection:* The detection of this behavior is similar to that of behavior  $BE2$ , except that behavior  $BM1$  appears in the middle stage. The LTL formula can be defined as follows.

```
#define p ((monitoredBid > (0.8*auc.reservePrice))
&&(monitoredBid <= auc.reservePrice))
#define q (monitoredBid > auc.reservePrice)
#define r (elapsedAuctionTime > middleStageStart)
<>(r && (<>p && (!<>q)))
```

*Analysis:* Let  $[0.8R, R]$  be a reserve price range, where  $R$  is the reserve price. With more bids in the reserve price range without exceeding  $R$ , a bidder can be more likely a shill because the bidder is trying to get the price as close to the reserve price as possible. In this case, the bidder should receive a high shilling score. This behavior can be further verified in the final stage. As described in Table 3,  $BF1$  says that when  $BM1$  or  $BM2$  is detected in the middle stage, if the bidder does not place larger bid than the reserve price, the bidder should receive more shilling points.

#### BF2: BM1 detected, and bids with small bid increments over the reserve price in the final stage

As an auction draws to the end, a shill bidder may still attempt to gain the highest possible price from other bidders. To avoid driving some other bidders away and accidentally winning the auction, a shill bidder uses a very small increment, as allowed by the auction house.

*Detection:* Detection of this behavior requires checking whether  $BM1$  has been detected, and whether small bid increments appear after  $BM1$  is detected, as well as identifying whether the monitored bid is greater than the reserve price or not.

```
#define p (monitoredInc < 10)
#define q (BM1.detected == 1)
#define r (monitoredBid > auc.reservePrice)
(<>q && (!!(q && r)) || <>((q && r) && <>p)))
```

*Analysis:* Detection of *BM1* is critical for detecting shilling behavior *BF2* because without behavior *BM1*, a bid with small bid increments over the reserve price in the final stage could be a very normal bidding behavior.

### 3.3 Algorithm for real-time shill detection

The monitoring agent has two major tasks. The first task is to detect shilling behaviors of a monitored bidder in a live online auction, which is supported by using the SPIN model checker. The second task is to assign shilling scores if any shilling behavior is detected. The major tasks of a monitoring agent can be illustrated by the following algorithm for real-time shill detection.

#### Algorithm: Real-Time Shill Detection

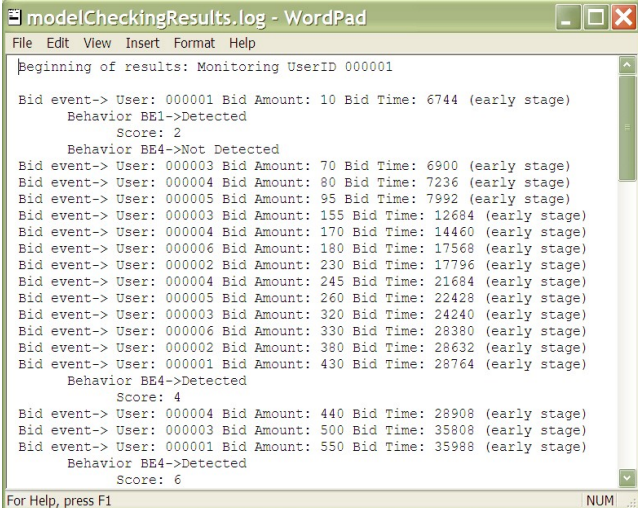
1. create an initial auction model for each involved auction
2. initialize shilling score  $ss = 0$  for monitored bidder  $mb$
3. set warning threshold  $wt$  and cancel-auction threshold  $ct$
4. **while** (any involved auction  $auc$  is active)
5.   **if** (*monitoredBidEvent* or *endOfStageEvent* occurs in  $auc$ )
6.     generateIncrementalModel ( $auc$ )
7.     DAM = CDAM  $\oplus$  IAM
8.     select a list of LTL formulas for current stage of  $auc$
9.   **for each** LTL formula for shilling behavior  $be$
10.     run SPIN model checker on DAM
11.     **if** (*valid*)
12.        $ss +=$  calculateShillingScore ( $be$ )
13.       **if** ( $ss > wt$ ) give warning to bidder  $mb$
14.       **else if** ( $ss > ct$ ) cancel auction  $auc$
15.     update CDAM with DAM for the next iteration
16.     save model checking results for IAM in next iteration
17. **else blocking**

As shown in the algorithm, the initial shilling score  $ss$  of a monitored bidder is set to 0. The monitoring agent continuously monitors a bidder as long as any of the involved auctions is still active. Once a *monitoredBidEvent* or *endOfStageEvent* occurs in auction  $auc$ , the DAM for  $auc$  is generated by combing the two models CDAM and IAM (denoted by operator  $\oplus$ ). A list of LTL formulas is selected for the current stage of  $auc$ , and each of them is checked using the SPIN model checker. If a certain shilling behavior is detected, a shilling score for that behavior is calculated and accumulated into the total shilling score  $ss$ . When  $ss$  exceeds the threshold for warning, the monitoring agent sends a warning message to the monitored bidder; on the other hand, if  $ss$  becomes larger than the threshold for canceling auction, all bidders on the involved auction will be notified, and the affected auction may be cancelled.

## 4 Experiments and analysis results

To demonstrate the efficiency and effectiveness of our approach, we implemented the real-time model checking module in a monitoring agent. We used some testing auction data generated using a recently implemented software bidding agent that supports specification of model-based bidding strategies [14]. The interface of the software bidding agent allows specification

of complex and flexible bidding strategies including normal and shilling ones. The auctioned item in our experiments is a bundle of NintendoWii, Playstation3, and Xbox 360 with an estimated price of \$1,500 and a reserve price of \$1,350. The duration of the agent-based online auction is 48 hours (i.e., 2 days). There are six agent bidders involved, namely Bidder 1 to Bidder 6 with user IDs from 000001 to 000006. Most of the bidders are normal bidders, which are specified with normal bidding strategies, but some bidders (i.e., Bidder 1 and 2) are specified with aggressive strategies that may involve certain shilling behaviors. The detailed procedure of specifying complex and flexible bidding strategy for bidding agents is beyond the scope of this paper, but can be found in our previous work [14]. We now simulate the auction data using CDAM and IAM that are dynamically generated in PROMELA as was described in Section 3.1, and let the model checking module automatically detect shilling behaviors in real-time. Figure 4 illustrates some model checking results for Bidder 1 (UserID: 000001) in the early stage of the agent-based online auction.



```

modelCheckingResults.log - WordPad
File Edit View Insert Format Help
Beginning of results: Monitoring UserID 000001
Bid event-> User: 000001 Bid Amount: 10 Bid Time: 6744 (early stage)
Behavior BE1->Detected
Score: 2
Behavior BE4->Not Detected
Bid event-> User: 000003 Bid Amount: 70 Bid Time: 6900 (early stage)
Bid event-> User: 000004 Bid Amount: 80 Bid Time: 7236 (early stage)
Bid event-> User: 000005 Bid Amount: 95 Bid Time: 7992 (early stage)
Bid event-> User: 000003 Bid Amount: 155 Bid Time: 12684 (early stage)
Bid event-> User: 000004 Bid Amount: 170 Bid Time: 14460 (early stage)
Bid event-> User: 000006 Bid Amount: 180 Bid Time: 17568 (early stage)
Bid event-> User: 000002 Bid Amount: 230 Bid Time: 17796 (early stage)
Bid event-> User: 000004 Bid Amount: 245 Bid Time: 21684 (early stage)
Bid event-> User: 000005 Bid Amount: 260 Bid Time: 22428 (early stage)
Bid event-> User: 000003 Bid Amount: 320 Bid Time: 24240 (early stage)
Bid event-> User: 000006 Bid Amount: 330 Bid Time: 28380 (early stage)
Bid event-> User: 000002 Bid Amount: 380 Bid Time: 28632 (early stage)
Bid event-> User: 000001 Bid Amount: 430 Bid Time: 28764 (early stage)
Behavior BE4->Detected
Score: 4
Bid event-> User: 000004 Bid Amount: 440 Bid Time: 28908 (early stage)
Bid event-> User: 000003 Bid Amount: 500 Bid Time: 35808 (early stage)
Bid event-> User: 000001 Bid Amount: 550 Bid Time: 35988 (early stage)
Behavior BE4->Detected
Score: 6
For Help, press F1 NUM

```

Figure 4. Model checking results for Bidder 1 (ID: 000001)

From Figure 4, we can see that shilling behavior *BE1* and *BE4* are detected for Bidder 1 in the early stage of the auction, and shilling scores are accumulated in real-time. Note that to simplify matters, we used a flat scoring strategy in our current implementation, where a constant value of 2 is added to the monitored bidder's accumulative shilling score each time a shilling behavior is detected. A more complex strategy for scoring would give a more accurate scoring mechanism, which will be implemented in a future version of the real-time model checking module.

Similarly, we run the model checking program for Bidder 2 and Bidder 3, and record the shilling scores of all three bidders over the auction time, as illustrated in Figure 5. Now if we set the threshold for warning as 15, Bidder 1 and 2 will receive a warning message around 18 and 35 hours after the auction starts, respectively (denoted by the two circles on the line with shilling score of 15 in Figure



5). If the threshold for cancelling an auction is set at 25, the auction will be cancelled around 2 hours before the auction ends (denoted by the circle on the line with shilling score of 25 in Figure 5). Since Bidder 3's shilling score never exceeds 15, we consider this bidder to be a normal bidder. Note that the determination of the thresholds are somehow subjective in this example; however, as the simulation results show, warnings are typically issued in the middle stage of an auction, while the action of cancelling an auction, which is a serious decision, shall be taken in the final stage of the auction.

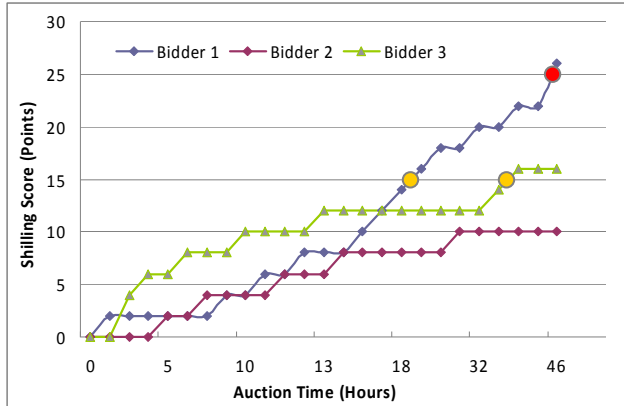


Figure 5. Comparison of shilling scores for three bidders.

## 5 Conclusions and future work

Shilling behaviors are becoming a more and more serious problem in online auctions, which may greatly damage the reputation of an auction house. In this paper, we present a real-time model checking approach to detecting shilling behaviors in live online auctions. Our approach supports creation of a dynamic auction model in real-time and timely detection of shilling behaviors in order to take appropriate actions. Our experimental results show that our approach is effective and efficient, and it is our vision that a real-time model checking module can be easily incorporated into the agent-based trust management (ATM) framework for trustworthy online auctions [11]. In future work, we will implement the ATM framework with the real-time model checking approach introduced in this paper. We also plan to extend our current approach to develop a concurrent dynamic auction model (CoDAM) for multiple online auctions, and demonstrate how a shill bidder can be detected in real-time when concurrent online auctions are involved.

## 6 References

- [1] D. H. Chau, S. Pandit, and C. Faloutsos, "Detecting Fraudulent Personalities in Networks of Online Auctioneers," In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Database (PKDD 2006)*, Berlin, Germany, September 2006, pp. 103-114.
- [2] H. Xu and Y-T. Cheng, "Model Checking Bidding Behaviors in Internet Concurrent Auctions," *International Journal of Computer Systems Science & Engineering (IJCSSE)*, July 2007, Vol. 22, No. 4, pp. 179-191.
- [3] G. J. Holzmann, "The Model Checker SPIN," *IEEE Transactions on Software Engineering*, Vol. 23, No. 5, 1997, pp. 279-295.
- [4] R. J. Kauffman and C. A. Wood, "Running up the Bid: Modeling Seller Opportunism in Internet Auctions," In *Proceedings of the Sixth Americas Conference on Information Systems (AMCIS 2000)*, M. Chung (Ed.), Long Beach, CA, 2000, pp. 929-935.
- [5] R. Bapna and A. Gupta, "Online Auctions: A Closer Look," In *The E-Business Handbook*, P. B. Lowry, R. J. Watson, and J. O. Cherrington (Eds.), St. Lucie Press, Boca Raton, FL, 2002, pp. 85-98.
- [6] C. Dellarocas and P. Resnick, "Online Reputation Mechanisms: A Roadmap for Future Research," *Summary Report of the First Interdisciplinary Symposium on Online Reputation Mechanisms*, April 26-27, 2003.
- [7] C. A. Wood, M. Fan, and Y. Tan, "An Examination of the Reputation Systems for Online Auctions", In *Proceedings of the Workshop for Information Systems and Economics (WISE 2002)*, Spain, December 2002.
- [8] S. Rubin, M. Christodorescu, V. Ganapathy, J. Giffin, N. Kidd, L. Kruger, and H. Wang "An Auctioning Reputation System Based on Anomaly Detection," In *Proceedings of the 12nd ACM Conference on Computer and Communication Security (CCS)*, Alexandria, VA, USA, November 2005.
- [9] V. Shmatikov and C. Talcott, "Reputation-Based Trust Management," *Journal of Computer Security*, Special Issue on Selected Papers of WITS 2003, Roberto Gorrieri (Ed.), Vol. 13, No. 1, 2005, pp. 167-190.
- [10] A. Boukerche and L. Xu, "An Agent-based Trust and Reputation Management Scheme for Wireless Sensor Networks," In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, Vol. 3, St. Louis, MO, November 2005, pp. 1857-1861.
- [11] H. Xu, S. M. Shatz, and C. K. Bates, "A Framework for Agent-Based Trust Management in Online Auctions," In *Proceedings of the 5th International Conference on Information Technology: New Generations (ITNG 2008)*, Las Vegas, NV, April 7-9, 2008, pp. 149-155.
- [12] J. Trevathan and W. Read, "Undesirable and Fraudulent Behaviour in Online Auctions," In *Proceedings of the International Conference on Security and Cryptography Conference*, Portugal, August 2006, pp. 450-458.
- [13] M. B. Dwyer, G. S. Avrunin, and J. C. Corbett, "Patterns in Property Specifications for Finite-State Verification," In *Proceedings of the 21st International Conference on Software Engineering (ICSE'99)*, Los Angeles, CA, 1999, pp. 16-22.
- [14] B. J. Ford, H. Xu, C. K. Bates, and S. M. Shatz, "Model-Based Specification of Flexible and Complex Bidding Strategies in Agent-Based Online Auctions," In *Proceedings of the 6th International Conference on Information Technology: New Generations (ITNG 2009)*, Las Vegas, NV, April 27-29, 2009.