A Software Tool for Collecting Data from Online Auctions

Rodel Balingit, Jarrod Trevathan, Yong Jin Lee and Wayne Read
Discipline of Information Technology
James Cook University, Australia
Email: rodel.balingit/jarrod.trevathan/yong.lee1/wayne.read@jcu.edu.au

Abstract— **Online auctions are increasingly becoming the platform of choice for dubious sellers to engage anonymously in fraudulent behaviour. While researchers are noble in their efforts to devise mechanisms to counter auction fraud, they are often frustrated by the lack of available auction data. Such data is an invaluable tool to gain insight about fraudulent traits and for testing proposed security remedies. This is compounded by online auction sources being non-cooperative in providing auction data, usually citing "security and privacy" as reasons for not wanting to help. This paper presents a software tool that can extract data from various online auction sources. The system is able to collect all the data for a given search criteria on auctions that have completed, and also returns later to collect the data from ongoing auctions once they have completed (without user intervention). We share our experiences from the development process and describe the challenges that must be overcome to successfully set up such a system. The data collected is used to analyse the behaviour and bidding patterns of sellers and buyers that are engaged in online auctions. The work presented in this paper represents the first serious attempt at creating an openly available software tool and establishing a repository of online auction data that will be free for use by other researchers.**

Keywords: e-Commerce, Software Design, Parsing, Statistics.

I. Introduction

Online auctions have become a popular place to conduct online trading. eBay[1], the world's largest online auction service is becoming more popular each day. Millions of people use eBay for its convenience and its ability to sell commonly available items and rare collectables not typically found in regular retail stores. However, the prospect of anonymity and lack of accountability in online auctions make it easy for individuals to engage in fraudulent behaviour. This behaviour might include artificially inflating the auction's price with fake bids (referred to as *shill bidding*), selling stolen items, or misrepresenting an item as being something that it is not.

Various techniques are now being designed to detect and/or prevent fraudulent behaviour (see [1], [5], [6], [9]). In order to enhance the effectiveness and the proposed mechanisms, researchers require real online auction data. This data can be used to analyse trends that allude to fraudulent behaviour and also to see the affect on buyers' and sellers' strategies given the presence of anti-fraud mechanisms. However, obtaining quality and volumness online auction data is quite a difficult task.

[1]http://www.ebay.com

Most commercial online auctioneers are unwilling to provide data and often cite security and privacy reasons. While it is true that security and privacy is a valid concern, the type of data required by researchers typically does not impede on these factors. Usually all that is required is the bidding history for a series of auctions. The bidders' and sellers' identities do not even need to be revealed. Instead it seems that the uncooperative nature of online auctioneers may be due to fear of lost business should it be discovered that fraudulent activity has taken place. By keeping the auction data private, auctioneers can conveniently keep any fraud that actually occurs from becoming public knowledge.

Researchers have used various techniques to obtain auction data without the help of online auctioneers. Some have manually "cut and pasted" data from auction pages (e.g., Jank and Shmueli [4]), whereas others automate the process using software tools (see Ruban et. al. [5], Shah et. al, [6]). Furthermore, some researchers have conducted their own auctions (see Trevathan and Read [9]), or used software bidding agents to artificially generate auction data (see Hattori et. al. [3], Trevathan and Read [8]). Regardless of the collection means, there appears to be no standard or documented manner in which the data collection has occurred. There also is no publicly available repository from which researchers can request data collected by other researchers.

This paper presents a software tool for extracting auction data from eBay. Once a user conducts a search on eBay, s/he enters the URL containing the auction listings. The system scans through listed auctions and extracts the item information and bid history. This tool is able to collect all the data for a given search criteria on auctions that have completed, and also returns later to collect the data from ongoing auctions once they have completed (without user intervention). The data is parsed and then stored in a database. The system also provides reporting features and statistics on the collected data. The work presented in this paper represents the first serious attempt at creating an openly available software tool and establishing a repository of online auction data that will be free for use by other researchers. (Note that in this paper we will be focusing on auctions run by eBay as an example auction site, however, the system can be adapted to extract auction data from other online auctioneers.)

This paper is organised as follows: Section II provides background on existing approaches that have been used to collect data from online auction sources. Section III describes the major software components of the system, the processes involved in collecting and storing the auction data, and also the reporting features to analyse the collected data. Section IV gives a basic performance analysis of the system and Section V describes some implementation specific issues. Section VI provides some concluding remarks and avenues for future work.

II. Existing Data Collection Approaches

This section discusses existing approaches and software tools used by researchers to collect data from various online auction sources.

Shah et. al. [6] developed a software tool called *Spider* that executes a search through the historical data of each product category. It collects the URLs to request individual auction data and the bidding history pages. The bid history page contains the details of submitted bids in a specific auction. The Spider caches both the auction details and bid history pages for each completed auction and parses them later. All requests are *staggered* to avoid putting too much load on eBay's server (i.e., the server is periodically polled). From the cached pages, the auction details were extracted and stored in a database.

Rubin et. al. [5] developed another type of software extraction tool. This tool procures basic information from an auction including the Seller's reputation according to eBay, and the bids that were placed. For each bid submitted, the amount, time, and bidder's username are recorded. The data is stored in a text file.

However, the problems with the aforementioned approaches by Shah et. al. [6] and Rubin et. al. [5] are that the software is not documented, nor is it publicly available for other researchers to use. This makes it difficult to perform comparisons between systems (approaches). Furthermore, these systems are now outdated due to the evolving structure of eBay auctions (e.g., masking bid IDs, private auctions, etc.).

A commercial tool referred to as *Auction Data Retriever*[2] exists that allows customers to harvest auction data directly from eBay web pages to their own customised auction management system. Customers can import the data to Microsoft Excel or Access. The fundamental problem with this system is that it is sales-driven in that it only retrieves basic sales information for a seller (but doesn't actually retrieve the bid history for any particular auction). Another deterring factor of this approach is that it costs money to subscribe to the system. Furthermore, there is no way to customise the type and amount of data collected. We therefore conclude that commercial systems aren't really conducive to conducting research into online auction fraud and security.

Trevathan and Read [7], [9] use a different approach to gathering auction data. They developed an online auction server that is capable of conducting real and simulated auctions. In this way they can directly obtain auction data from the participants. However, the problem with this approach is that it takes a significant amount of time to set up, promote and oversee the auctions. Essentially the researcher must become a would-be auctioneer. It is also difficult to gather enough willing people to cooperate in auctions in order to obtain meaningful data.

[2]http://rajeware.com/

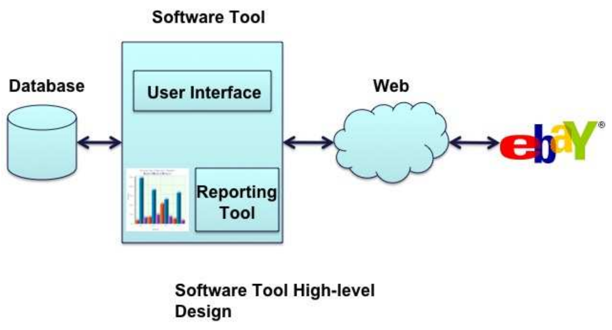


Fig. 1. The High-Level Software Model for Collecting Online Auction Data

To get around the aforementioned problem, the auction server (and an approach outlined in Hattori et. al. [3]) also allowed for software bidding agents to artificially generate data (see Trevathan and Read [9]). While this is useful, software bidding agents still don't adequately capture the real-world behaviour of human beings. Therefore generated auction data has limited usefulness with regard to analysing fraudulent trends and testing the effectiveness of anti-fraud techniques.

III. Software Model and Design

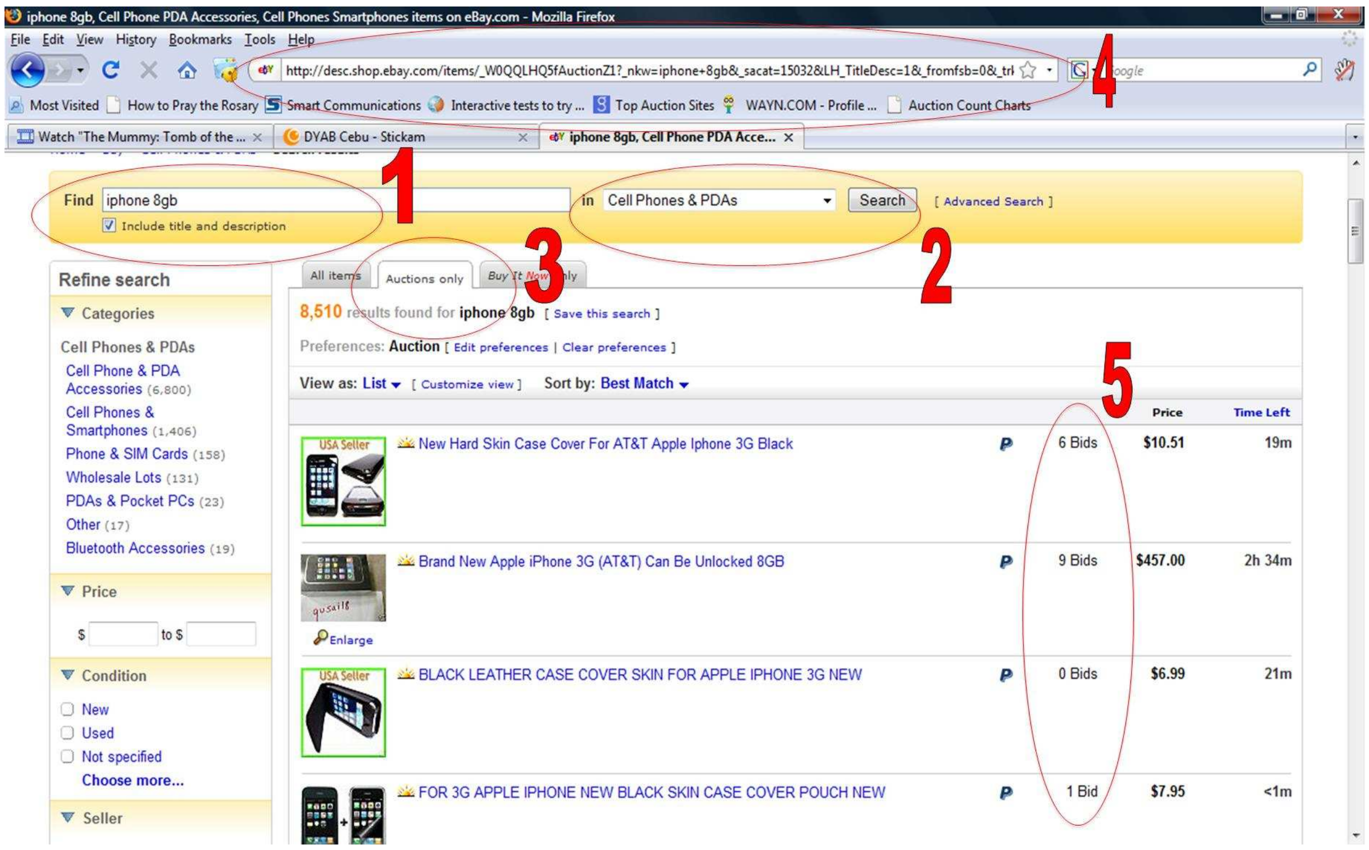
This section presents the software model and design of the auction data extraction tool. It describes the process of collecting the auction data, the internal components, storage of the data and reporting capabilities.

A. High Level Design

Figure 1 presents the high level software design for the auction data extraction tool. Essentially the system sits on a user's computer and interacts with eBay across the web. Once started, the tool is continually online until all auctions have completed and their information has been extracted. If an auction has not completed when initially observed, the system records the finish time and then revisits the auction once it has completed to extract its final data. The results (i.e., the auction data) are written to a database. At any time the user can interact with the software by a user interface. Once the extraction process commences, the user can leave the system running as a batch process. The system can also write data to a database or an Excel file for statistical analysis.

B. Setting Up for Data Collection

Before a software tool is run, a user must initially visit eBay's auction site and undertake a search for the auctions s/he desires to extract data from. Once the user has refined his/her search, s/he uses the URL of the page as input into the extraction tool. The tool uses the URL as the initial starting point to commence the data extraction process. Note that this



http://desc.shop.ebay.com/items/_W0QQLHQ5fAuctionZ1?_nkw=iphone+8gb&_sacat=15032&LH_TitleDesc=1&_fromfsb=0&_trk

Find: iphone 8gb in: Cell Phones & PDAs

☑ Include title and description

8,510 results found for **iphone 8gb**

Preferences: Auction

Cell Phones & PDAs
- Cell Phone & PDA Accessories (6,800)
- Cell Phones & Smartphones (1,406)
- Phone & SIM Cards (158)
- Wholesale Lots (131)
- PDAs & Pocket PCs (23)
- Other (17)
- Bluetooth Accessories (19)

Price: $ to $

		Price	Time Left
New Hard Skin Case Cover For AT&T Apple Iphone 3G Black	6 Bids	$10.51	19m
Brand New Apple iPhone 3G (AT&T) Can Be Unlocked 8GB	9 Bids	$457.00	2h 34m
BLACK LEATHER CASE COVER SKIN FOR APPLE IPHONE 3G NEW	0 Bids	$6.99	21m
FOR 3G APPLE IPHONE NEW BLACK SKIN CASE COVER POUCH NEW	1 Bid	$7.95	<1m

Fig. 2. An Example of How to Set the System Up for Data Collection

is the only part of the process that requires manual intervention by a user.

The basic steps involved for initialising the system are as follows (refer to Figure 2):

1) The user visits the eBay auction site and searches for a desired item on auction (e.g., *Playstation 3 (PS3), Nokia N95 8GB phone, iPhone*).
2) The user selects the *All Categories* drop down menu to refine search lists of *Individual Auction Items* and then clicks the "Search" button.
3) The user selects the *Auction only* tab to display the items on auction only.
4) The user must copy the URL found on the browser's address bar.
5) The user can verify that s/he has refined the search correctly by checking that bids have been submitted for an individual auction. This will indicate that the tool will extract data from the correct auction format (refer to Section III-C).

While, the search can be done in different ways to that described above, the objective here is to get the correct URL value from the browser's address bar as input to the system.

C. The Data Collection Process

eBay offers differing buying formats for buyers and sellers to conduct their business. These buying formats can be categorised as:

- **Auction** – This format is an interaction of bidding among competitive buyers. It has four types: *Auction-like listings*, *Second Chance Offer*, *Multiple Item Auction* and *Special Sites (i.e., eBay Live auction, eBay Motors)*.
- **Fixed Price** – This format is offered to a buyer who wants to pay a fixed price and get the item quickly.
- **Advertisement** – This format does not directly list the item for sale, instead, it helps to connect buyers with sellers (e.g., *Want It Now* and *Best Offer* auctions) (see [2]). This format also encompasses the businesses affiliated with eBay which offer goods directly to the customer via eBay.

The software tool only extracts data from *Auction* listings. At any point in time the tool categorises an auction as either an *ongoing auction* or a *completed auction*. Generally, the software extraction tool collects data from an ongoing auction by individually parsing the *Auction Items page* list, then parses/extracts the *Individual Auction Item* details and bid history information. The software parsing tool can perform an update on a recently completed auction automatically while the software tool is busy extracting data from other auctions.

Figure 3 illustrates the process by which the tool collects auction data from eBay. Once the system has been initialised (i.e., set to the search page), the software tool constructs a *URLAuctionList page*. This is done by extracting the URLs

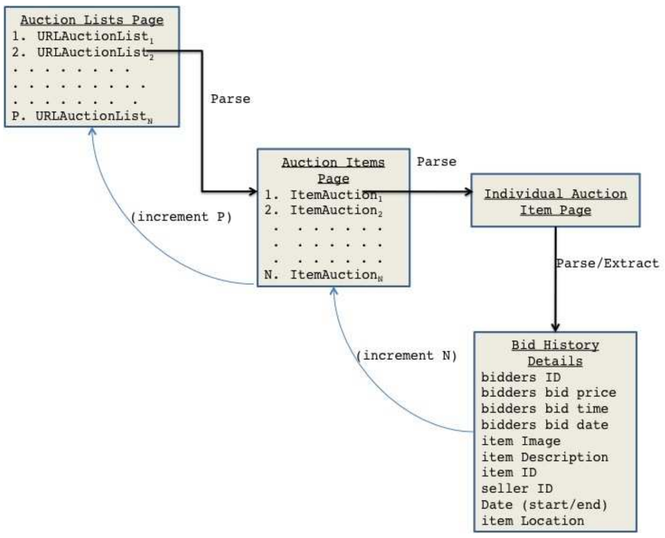


Fig. 3. The Process Involved with Collecting Data from an Online Auction Source

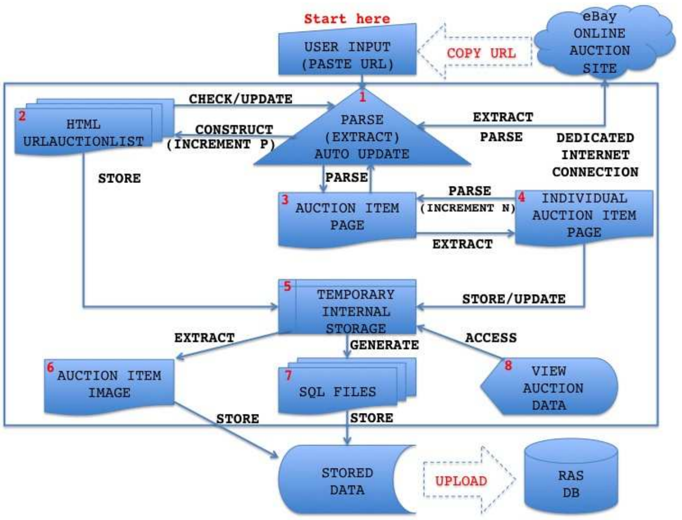


Fig. 4. The Online Auction Data Software Extraction Tool Design

for each auction from the raw HTML in the search page.

For each URL page, the *Individual Auction Item* list page is parsed and the following information is extracted:

- Item ID
- Seller ID
- Item image
- Item description
- Date (start/end)
- Item location

Next, the link to the item's *Bid History* is followed and the following information is extracted for each bid submitted:

- Bidder's ID
- Price
- Time and date submitted

This data is cached until the last *Individual Auction Item* on that page is completely parsed and extracted. The cached data is written individually to a text file and then moves to the next auction in the *URLAuctionList page* list. This process is repeated until the last *URLAuctionList page* is reached.

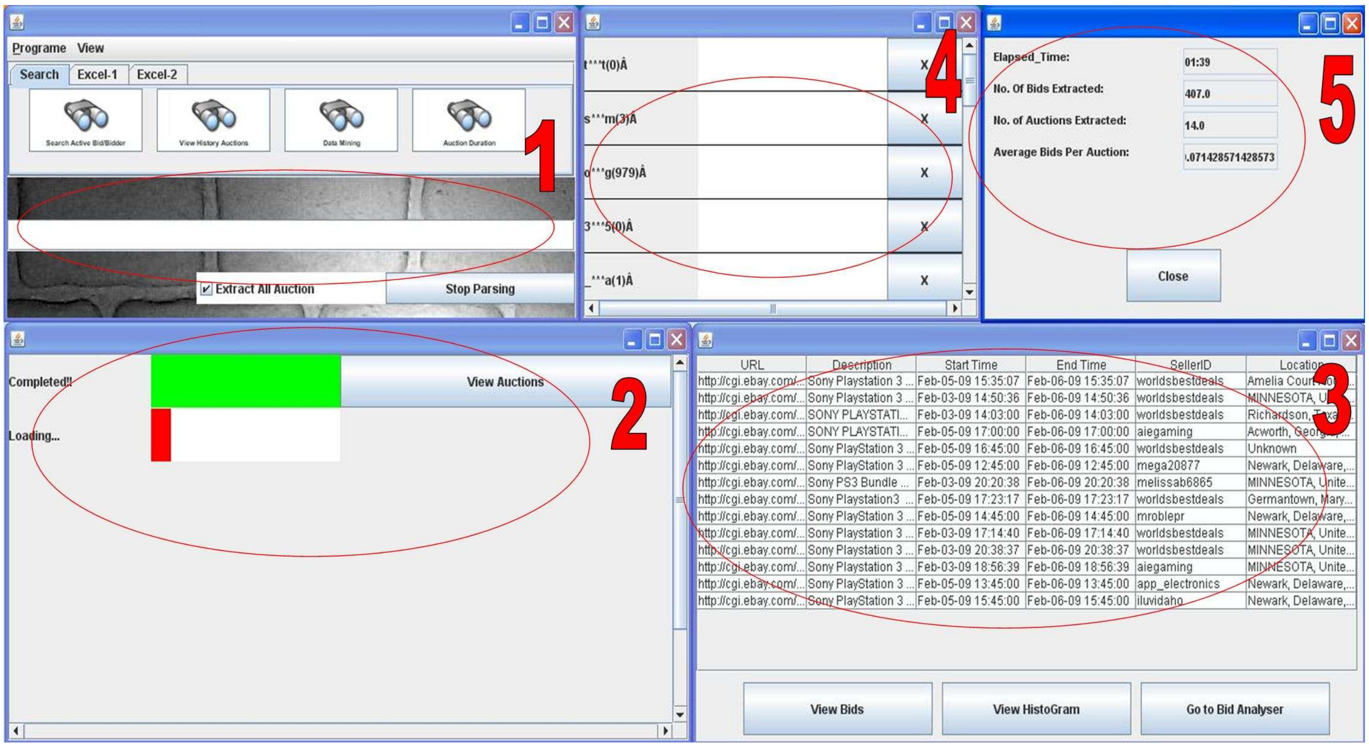
D. Database Storage

The auction data collected is prepared and written in a form of a SQL command stored procedure file. This allows the data to be directly inserted into any database conforming to a schema similar to the Research Auction Server (RAS). RAS is an academic auction server designed to conduct both simulated and real auctions for the purposes of collecting auction data. Refer to Trevathan and Read [7] for the precise definition of RAS's database.

E. Internal Functional Components

Figure 4 illustrates the internal architecture of the processes involved for extracting and storing the auction data. There are several main components:

1) *PARSE/EXTRACT/AUTO UPDATE* (PEAU) – This parses eBay's auction HTML documents, extracts auction data for each item on auction, and performs an auto update of a recently completed auction. Both the parse and extract processes are performed simultaneously. During this process, PEAU can detect if a specific auction has completed, and then automatically updates the previously extracted auction data. The auction data is cached during the parse/extract process and written to *TEMPORARY INTERNAL STORAGE*.
2) *HTML URLAUCTIONLIST* (HURL) – This holds a constructed *URLAuctionList page* during the parse/extract process. The *URLAuctionList page* is appended with additional *URLAuctionList* data as the parse/extract process moves to the next page (*increment P* in Figure 3) until the operation is completed. The constructed list is used as reference to monitor and update the extracted auction data from recently completed auctions.
3) *AUCTION ITEM PAGE* (AuIP) – This holds a list of items on an auction as per the *URLAuctionList page*. Each item parsed is checked to determine if submitted bids are found. Then, it moves to the IvAuIP, otherwise it continues to parse the next auctioned item (*increment N* in Figure 3), or goes to the next *URLAuctionList page* as the last auctioned item is reached.
4) *INDIVIDUAL AUCTION ITEM PAGE* (IvAuIP) – An IvAuIP holds the bid history details. At this point, an extraction of data commences and is then stored in *TEMPORARY INTERNAL STORAGE*.
5) *TEMPORARY INTERNAL STORAGE* – This is a central repository of an extracted auction data from eBay. The data stored are organised and prepared for file creation like *url.txt*, *auction.sql*, *bidinfor.sql*, *updatebidinfor.sql*, and image file. The *url.txt* file is a list of URLs collected from eBay. The files *auction, bidinfor and updatebidinfor* are SQL command procedures used for inserting



t***t(0)Â	X
s***m(3)Â	X
o***g(979)Â	X
3***5(0)Â	X
_***a(1)Â	X

Elapsed_Time:	01:39
No. Of Bids Extracted:	407.0
No. of Auctions Extracted:	14.0
Average Bids Per Auction:	.071428571428573

URL	Description	Start Time	End Time	SellerID	Location
http://cgi.ebay.com/...	Sony Playstation 3 ...	Feb-05-09 15:35:07	Feb-06-09 15:35:07	worldsbestdeals	Amelia Court Ho...
http://cgi.ebay.com/...	Sony Playstation 3 ...	Feb-03-09 14:50:36	Feb-06-09 14:50:36	worldsbestdeals	MINNESOTA, U...
http://cgi.ebay.com/...	SONY PLAYSTATI...	Feb-03-09 14:03:00	Feb-06-09 14:03:00	worldsbestdeals	Richardson, Texa...
http://cgi.ebay.com/...	SONY PLAYSTATI...	Feb-05-09 17:00:00	Feb-06-09 17:00:00	aiegaming	Acworth, Georgia, ...
http://cgi.ebay.com/...	Sony PlayStation 3 ...	Feb-05-09 16:45:00	Feb-06-09 16:45:00	worldsbestdeals	Unknown
http://cgi.ebay.com/...	Sony PlayStation 3 ...	Feb-05-09 12:45:00	Feb-06-09 12:45:00	mega20877	Newark, Delaware,...
http://cgi.ebay.com/...	Sony PS3 Bundle ...	Feb-03-09 20:20:38	Feb-06-09 20:20:38	melissab6865	MINNESOTA, Unite...
http://cgi.ebay.com/...	Sony Playstation3 ...	Feb-05-09 17:23:17	Feb-06-09 17:23:17	worldsbestdeals	Germantown, Mary...
http://cgi.ebay.com/...	Sony PlayStation 3 ...	Feb-05-09 14:45:00	Feb-06-09 14:45:00	mroblepr	Newark, Delaware,...
http://cgi.ebay.com/...	Sony Playstation 3 ...	Feb-03-09 17:14:40	Feb-06-09 17:14:40	worldsbestdeals	MINNESOTA, Unite...
http://cgi.ebay.com/...	Sony Playstation 3 ...	Feb-03-09 20:38:37	Feb-06-09 20:38:37	worldsbestdeals	MINNESOTA, Unite...
http://cgi.ebay.com/...	Sony Playstation 3 ...	Feb-03-09 18:56:39	Feb-06-09 18:56:39	aiegaming	MINNESOTA, Unite...
http://cgi.ebay.com/...	Sony PlayStation 3 ...	Feb-05-09 13:45:00	Feb-06-09 13:45:00	app_electronics	Newark, Delaware,...
http://cgi.ebay.com/...	Sony PlayStation 3 ...	Feb-05-09 15:45:00	Feb-06-09 15:45:00	iluvidaho	Newark, Delaware,...

Fig. 5. An example summary of auctions which are being extracted from eBay (showing both in-progress and recently completed auctions). 1) illustrates where the URL for the auction search page is entered. 2) indicates how far through the entire extraction process the tool currently is. 3) shows a summary of the auctions from which data is being collected. 4) presents the bid history data collected from a specific auction. 5) provides summary statistics for the auction data collected.

the data directly into the database.

6) *AUCTION ITEM IMAGE* (AII) – This extracts the images for each item auctioned.
7) *SQL QUERY FILES* – This creates a SQL stored procedure to insert extracted auction data into a RAS-style auction database. It creates three files: *auction.sql*, *bidinfor.sql* and *updatebidinfor.sql*.
8) *VIEW AUCTION DATA* (VAD) – This allows a user to view the extracted auction data. The view feature includes auction data, URL list and updates of recently completed auctions.

F. Reporting Features

The auction data extraction tool can perform a preliminary statistical analysis and provides a histogram/graph to initially analyse the extracted auction data.

Figure 5 shows an example of the software parsing tool's processed output. The five separate windows are displayed when the software tool starts the auction data extraction process. First, the copied URL address is placed in the input text field by a user initially (refer to *Window 1* of Figure 5). The user has the option to tick the "Extract All Auction" checkbox and then selects the "Start Parsing" button to execute the process. The advantage of this feature is that the software parsing tool can perform an auto increment to proceed to the next *URLAuctionList page* to undertake the parse/extract process until the last page is reached. Otherwise, only the current *URLAuctionList page* is parsed/extracted.

Window 2 of Figure 5 shows the overall progress made on extracting the auction data. It lists the number of completed auctions that have been processed and the number of ongoing auctions that the system is still waiting on to finalise data extraction. If the user clicks the "View Auctions" button, *Window 3* appears (see Figure 5). This window presents the details for all of the auctions that will have their data extracted. *Window 4* allows the user to view to bid history that has been extracted for a specific auction. *Window 5* presents statistics for the software tool pertaining to the total elapsed time since the tool commenced execution, the number of bids extracted, the number of auctions extracted, and the average number of bids per auction that have been extracted.

IV. Complexity Analysis

This section performs a basic complexity analysis of the auction data extraction software.

The factors that influence the system's running time include: 1) the number of auctions to extract; and 2) the finishing times of the auctions. Assuming that all auctions have completed, the execution time is linear in relation to the number of auctions. That is, given n auctions, the running time is $O(n)$.

To extract auction URLs from the initial search page, the system must open the HTML file and sequentially scan the HTML tags to remove the relevant data. This process is also similar for extracting the auction data and bidding history for each auction.

Since the system extracts data from ongoing and completed auctions, when it encounters an ongoing auction, it must later return to that auction to extract its final data. As previously mentioned, the system does this by recording the auction's closing time. It then returns when the user's computer's system clock indicates that the time on the auction server has passed the auction's closing time. Using this approach, the system does not need to continually poll the auction server (as in the approach by Shah et. al. [6]). The only load the system places on the auction server is a one off extraction of the search page and each relevant auction page. Given n auctions with m ongoing auctions (where $m \leq n$), the system places a maximum load on the auction server of $O(n+m)$ (in addition to the extraction of URLs from the search page).

V. Implementation

This section briefly describes the specifics of the implementation and practical issues we faced with using the system.

The auction data extraction tool is developed and implemented using the Java programming language. Several versions were implemented and unit tested prior to the commencement of a concerted acquisition attempt to obtain data. We ran the software tool on several standard desktop computers simultaneously to extract auction data. Each computer extracted data from different search criterias for different auctions. At the end of the process, all of the auction data was collated and stored in a central repository.

At the time of writing, the system has collected data from over 1300 auctions spanning three categories of items. The data has taken one month to collect. There were several major impediments to the time it took to collect this data. The first constraint is the number of auctions available for the desired search criteria. Next the system is constrained by the duration of each of the auctions. For example, some auctions can run for a day, whereas others may run for ten days. Obviously more data can be collected for a large number of auctions that run for a short duration.

A significant limiting factor specific to our data collection was a proxy authentication system within our research institution. To gain external Internet access, each user (or process) must authenticate him/herself to the proxy. The authentication remains active for twelve hours of continuous use before requiring reauthentication. This essentially denied the system external access to eBay when performing a batch job (i.e., it exceeded the twelve hours of activity), thereby forcing the system to have to be restarted. We eventually overcome this by allowing the system to reinitialise and take up where it left off from disconnections.

VI. Conclusions

This paper presented a software tool to collect data from eBay's new auction structure. This work is motivated by the unwillingness of commercial auctioneers to contribute auction data for the purposes of testing auction fraud detection/prevention mechanisms that are proposed by researchers. The software tool parses/extracts auction data from both ongoing and completed auctions, and collects description information pertaining to the item auctioned as well as its bidding history. The collected data can be exported to an Excel spread sheet to graph for analysis. Additionally, the auction data collected can be uploaded to a customised auction server. The work presented in this paper represents the first serious attempt at creating an openly available software tool and establishing a repository of online auction data that will be free for use by other researchers.

Future work involves opening the software tool to collaboration with other researchers to improve its performance (e.g., increase its efficiency to extract auction data, revise the software model and architecture design, etc.). The database of collected data will be put online for other researchers to copy and/or add their own extracted auction data to. We also intend on allowing the software tool to extract all the completed auctions for a specific seller, and adding additional statisitical functionality. Furthermore, there is scope to implement basic data mining algorithms in an attempt to analyse previously unknown trends in the auction data over time.

References

[1] Y. Cheng and H. Xu, "A Formal Approach to Detecting Shilling Behaviours in Concurrent Online Auctions," in *Proceedings of the* 8^{th} *International Conference on Enterprise Information Systems*, 2006.

[2] eBay, "Guide to Buying Formats", Online, http://pages.ebay.com.au/help/buy/formats-ov.html, 04 August 2008.

[3] H. Hattori, R. Yamada, T. Ozono and T. Shintani, "A Multiple-Bidding Support Framework for Bidding and Browsing Information" - *Technical Report*, Department of Intelligence and Computer Science Nagoya Institute of Technology, 2002.

[4] Jank and Shmueli. Dynamic Profiling of Online Auctions Using Curve Clustering. In *Proceedings of the* 11^{th} *Annual Spring Research Conference (SRC) on Statistics in Industry and Technology*, 2004.

[5] S. Rubin, M. Christodorescu, V. Ganapathy, J. Giffin, L. Kouger and H. Wang, "An Auctioning Reputation System Based on Anomaly Detection", in the *Proceedings of the* 12^{th} *ACM Conference on Computer and Communications Security (CCS)*, pages 270–279, Alexandria Virginia, 2005.

[6] H. Shah, N. Joshi and P. Wurman "Mining for Bidding Strategies on eBay", in *SIGKDD'2002 Workshop on Web Mining for Usage Patterns and User Profiles*, 2002.

[7] J. Trevathan and W. Read, RAS: a system for supporting research in online auctions, *ACM Crossroads*, *12.4*, 23–30, 2006.

[8] J. Trevathan and W. Read, "Detecting Collusive Shill Bidding," in the *Proceedings of the* 4^{th} *International Conference on Information Technology - New Generations*, pages 799–808, 2007.

[9] J. Trevathan and W. Read "Detecting Shill Bidding in Online English Auctions," *Social and Human Elements of Information Security: Emerging Trends and Countermeasures*, IGI Press, pages 446–470, 2008.