

# Personalized Bidding Agents for Online Auctions

*Junling Hu*

School of Business, University of Mississippi

junling@olemiss.edu

<http://www.bus.olemiss.edu/jhu/index.html>

*Daniel Reeves      Hock-Shan Wong*

AI Lab, University of Michigan

{dreeves,hs Wong}@umich.edu

<http://ai.eecs.umich.edu/people/{dreeves,hs Wong}>

## **Abstract**

We have designed configurable bidding agents to represent users in one specific online auction: the Michigan AuctionBot. The agents can be configured, started, and monitored from a web interface. We implemented three types of bidding agents, distinguished by their different ways of using information in the auctions. A competitive agent does not use any information in the auction market. It chooses its actions based on its individual optimization problem. A price modeling agent uses price history as its only information. A bidder-modeling agent uses other agents' bidding histories to predict their next bids and infer the next clearing price. Our experiments suggest that an agent's performance in the auctions depends not only on its bidding strategy, but also on the bidding strategies of others.

## **1 Introduction**

The study of intelligent agents for electronic commerce has become popular nowadays. We have seen shopping agents that collect price information for

users [4, 2], and information filtering agents that collect interesting publications [1]. Our particular interest is in agents for online auctions. Such agents can bid on the behalf of users. The need for such agents arises from the fact that users may not have time or inclination to monitor the activities in an auction. Sometimes an optimal bidding strategy may be computationally intensive, in which case it is especially useful to have a software agent carry out the bidding.

Current auction agents on the Internet are mainly auction search agents. The Bidder's Edge agent ([www.biddersedge.com](http://www.biddersedge.com)) monitors ongoing auctions and watches for items in upcoming auctions. The agent on Tripod Auctions [15] constantly browses the listings to find a customer's favorite items for sale. It notifies the customer when an auction is placed that matches the customer's criteria. What are missing from these agents are the capabilities of submitting intelligent bids for customers.

The ability of submitting intelligent bids requires an agent to make rational decision. The agent has to be able to take advantage of the information available and choose the bids that will maximize its profit over the long run. We are particularly interested in the issues that other bidders on the auctions may bid intelligently at the same time. The optimal strategy of our agent may crucially depend on how other bidders react. Researches have been conducted on intelligent agents in auctions, but such agents have not been deployed in the Internet environment. We not only provide such agents for online auctions, but also implement the multi-agent learning capabilities in the agents.

We have designed an agent server that works on a user's behalf to submit bids to one specific online auction—the Michigan AuctionBot. The users specify the names of the auctions they want to participate in, the initial amount of the goods, and the bidding strategies they prefer. The agent server then invokes agent programs to bid on the AuctionBot for the users. The agents keep bidding until the auction closes, and then report the results back to the users.

We implemented three types of bidding agents, distinguished by their different ways of using information in the auctions. A competitive agent does not use any information in the auction market. It chooses its actions based on its individual optimization problem. A price-modeling agent uses price history as its only information. A bidder-modeling agent uses other agents' bidding histories to predict their next bids and infer the next clearing price. Our experiments suggest that an agent's performance in the auctions

depends not only on its bidding strategy, but also on the bidding strategies of others. It remains an open question as what constitute a best bidding strategy.

## **2 Design Overview**

### **2.1 Michigan AuctionBot**

The Michigan AuctionBot [19] is a configurable auction server. It allows human agents to create auctions and submit bids via web forms, and software agents to perform the same operations via TCP/IP. This auction server has been operational since September 1996. Currently, the Michigan AuctionBot supports many auction types including English auctions, Dutch auctions, and Vickrey auctions. These different auctions are distinguished by the way bidders submit bids and how the allocations and prices are determined [10]. As far as we know, the Michigan AuctionBot is the only auction site that provides an API to enable software agents to directly talk to the server.

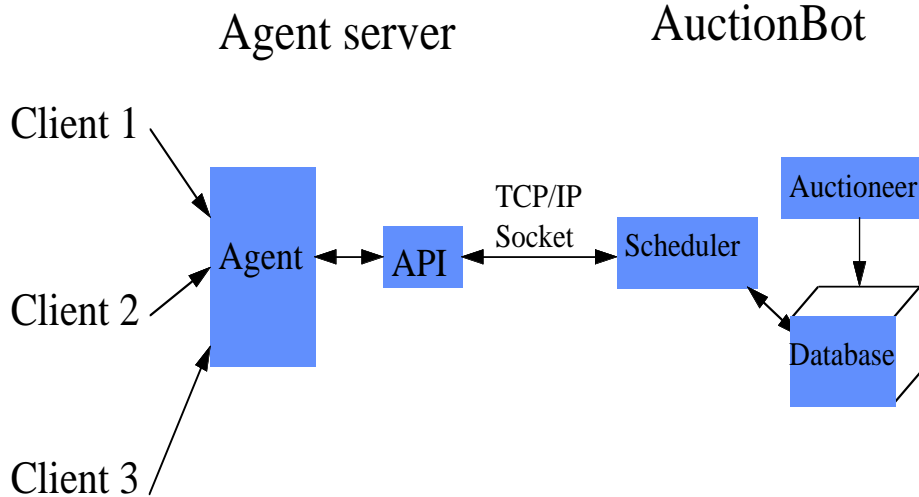
The AuctionBot API [11] is a client-server communication protocol that is straightforward to implement for client developers in many languages and on many different platforms. The AuctionBot API functions reside on a server. Interfaces to the functions are well-defined messages encoded as strings that are sent to the server through a socket and invoke the API functions that run on the server. The server functions return string-based messages through the socket to the API client, informing it of the results of the request.

### **2.2 Agent Server**

#### **2.2.1 Overview**

Our agent server interacts with users through an HTML form. The underlying agent programs are invoked through CGI script. The agent programs are run in C++ and Mathematica code. The agents will then try to establish the TCP/IP connection and authenticate with the AuctionBot using the AuctionBot's API. After a connection is established, the agent starts submitting bids for the user. The agent keeps track of the bid status and submits subsequent bids after each clear of the auction. This process continues until the auction closes or the user stops the agent from running.

Figure 1: Agent server overview



### 2.2.2 User Interface

Our agent server provides an easy-to-use web interface to assist users. First-time users are required to register before using the service. The registration verifies that the user has a valid account with the Michigan AuctionBot. The user's ID, encrypted password, and email address are stored in a protected directory. For security purposes, users have to authenticate themselves every time they request an agent service.

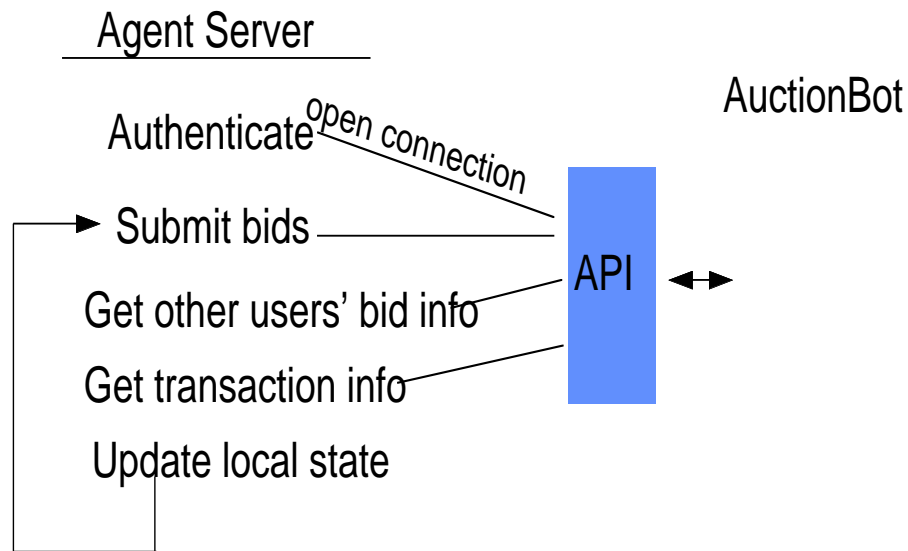
When users request an agent bidding service, they can specify one of three types of agents provided by the service. The details of these types are discussed in later sections. Users also specify the initial endowments for the two types of goods. The user's utility is computed as a function of these endowments. This utility function is defined by the agent service based on certain parameters.

As the agent runs, it updates the user's browser dynamically by maintaining a persistent HTTP connection. New data is sent to the client until either the AuctionBot server or the client stops the connection. The continuous updates allow the user to keep track of the bidding process dynamically.

## 2.3 Communication with AuctionBot

In order to communicate with the AuctionBot, our agent has to open a TCP/IP connection and then authenticate itself by submitting its user ID. All the communication is made through the API protocol. The agent can then submit bids, get information on other agents' bids, find out about clearing prices, and determine the transactions it has been involved in.

Figure 2: Communication with AuctionBot



For example, to find out about its transactions, the agent sends the string

```
transid?order="time"
```

This returns a list of transaction IDs, ordered by time, such as

```
transid?id=123&id=124&id=125...
```

The agent then parses this result to get the ID of the latest transaction, say 125, and asks for more detailed information by sending

```
transinfo?transid=125
```

to which AuctionBot responds

```
transinfo?id=125&auction=789&cleartime=
886015200&buyer=782&seller=121&price=3.14
&quantity=1&status=0
```

This means that in auction number 789 which cleared at the specified time, one unit was bought for \$3.14 between user 782 and 121. The status of zero indicates no error.

An auction goes through many rounds. At each round, agents submit their bids. The auction server then finds a clearing price, and matches the sellers and buyers at that price. Each pair of buyer and seller transacts one unit of good at the clearing price. The results are sent back to the agent. Then the auction goes into the next round. Figure 2 illustrates this process.

The most difficult design issues in the communication with the Auction-Bot are timing issues and network delays. Since the auction that the agents are bidding in is synchronized, the agents need to submit their bids and then check the bids of other agents before the auction clears. After waiting for the clear, they check to see if they transacted and update their endowments and utilities accordingly before submitting another set of bids and repeating the cycle.

### 3 The auction environment

Different types of auctions involve different bidding strategies. We studied bidding strategies for one type of auctions: the *Mth-Price Auction* [18], where the clearing price is set at the  $M$ th highest among all bids and  $M$  ( $M \geq 1$ ) is the number of sell bids. The  $M$ th-price auction is a subclass of double auctions. In order to understand  $M$ th-Price Auctions, we have to understand double auctions.

#### 3.1 Double auctions

In a *double auction* [7], there are multiple buyers and multiple sellers. An agent may submit both buy bids and sell bids. A typical double auction has the following features: (1) One unit of a good is traded each time period; (2) Bids are observable to all agents once they are submitted; (3) Each agent's preferences are unknown to other agents.

Based on the timing of the bidding protocol, double auctions can be classified as *synchronous* (or *synchronized*) or *asynchronous* double auctions.

In a synchronized double auction, all agents submit their bids in lockstep. Bids are “batched” during the trading period, and then cleared at the end of the period. In an asynchronous double auction, also called a *continuous* double auction, agents offer to buy or sell and accept other agents’ offers at any moment. Continuous double auctions have been widely used in stock exchange markets [6] and Internet auctions. Synchronized double auction can be seen in Nasdaq and clearing houses.

The book edited by Friedman and Rust [7] collects several studies of double auctions, including both simulations and game-theoretic analyses. Game-theoretic studies [14] [6] on double auctions generally adopt the framework of static (one-shot) games with incomplete information, for which the equilibrium solution is Bayesian Nash equilibrium. Double auctions are essentially dynamic games. Since agent interaction takes more than one round, the static game framework fails to address the basic dynamics of the system. Other theoretical studies [5] try to explain the experimental data generated from human subjects. While the study of human behavior is interesting, we are more interested in designing artificial agents who can bid as intelligently as possible to get maximum payoffs.

### 3.2 Agent design for double auctions

Gode and Sunder [8] designed *zero-intelligence* agents who submit random bids within a range such that their utilities never decrease. To improve upon zero-intelligence agents, Cliff [3] designed *zero-intelligence-plus* agents who submit bids within the utility increasing range, but the bids are chosen so that their utilities will increase by some proportion which is adjusted over time. The effectiveness of the learning depends on several parameters including the learning rate. Cliff implemented a genetic algorithm (GA) to let the agent learn about these parameters. The training for the GA requires the agent to know the final convergence price of the whole auction. It is not clear how such GA training can be applied to online settings.

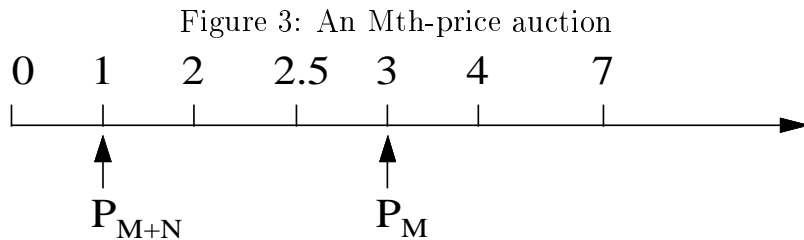
Other types of intelligent agents have also been designed for double auctions. Park et al [12] designed an adaptive p-strategy agent to participate in continuous double auctions. Her experiments showed that the p-strategy out-performed other strategies. In the Santa Fe Tournament [13], 30 different intelligent programs competed in a synchronized double auction. A simple non-adaptive agent won that tournament by always waiting in the background and letting the others do the negotiation. When the bid and ask

prices were sufficiently close the agent jumped in and stole the deal. Such an agent is not applicable when there is no negotiation process in the auction. Hu and Wellman [9] has designed an agent that learns other bidders' behavior and choose its bids as a best respond to others' potential bids. Their methodology has been directly applied to this paper.

### 3.3 Bidder Matching in Mth-Price Auctions

A buyer who submits a bid above the clearing price can be matched to a seller with a bid below the clearing price. For Mth-price auction, it has been proved that all buy bids above the clearing price can be matched to all sell bids below the clearing price [18]. The buy bids below the clearing price cannot be matched. The same thing applies to the sell bids "above the clearing price.

An example is given in Figure 3. Suppose the buy bids are  $\{2, 4, 7\}$  and the sell bids are  $\{1, 2.5, 3\}$ . The clearing price is therefore 3, which is the third highest price. Given this clearing price, we can see that any of the two sell bids 1 and 2.5 can be matedched to any of the two buy bids 4 and 7. The buy bid 2 and sell bid 3 can not be matched to each other or to any of the other bids.



### 3.4 Agent model

We assume that our agent has a utility function

$$U(x) = \left( \sum_g (x_g)^{\frac{1}{2}} \right)^2, \quad (1)$$

where  $x = (x_1, \dots, x_m)$  is a vector of goods. This function is a special case of CES (Constant Elasticity of Substitution) utility function  $(\sum \alpha_g (x_g)^\rho)^{\frac{1}{\rho}}$  [16],

where the  $\alpha_g$  are preference weights, and  $\rho$  is the substitution parameter. When  $\alpha_g = 1$  for all  $g$  and  $\rho = \frac{1}{2}$ , the utility function becomes (1).

The reward for the agent at time  $t$  is given by

$$\begin{aligned} r_t &= U_{t+1} - U_t \\ &= U(x_{t+1}) - U(x_t) \end{aligned}$$

which is the increment in utility through the change of the vector of goods. Since no good comes for free, the agent has to reduce one type of good in order to increase the holding of another type of good.

In constructing agent strategies, we dictate that an agent always chooses a bid that its reward is nonnegative. A price that makes the agent's utility the same as before (so that the reward is zero) is called the agents' *reservation prices* [16]. Let  $\bar{P}_b$  and  $\bar{P}_s$  be an agent  $i$ 's reservation buy and sell price. They satisfy the following conditions:

$$U(x_g - 1, x_m + \bar{P}_s) = U(x_g, x_m), \quad (2)$$

$$U(x_g + 1, x_m - \bar{P}_b) = U(x_g, x_m). \quad (3)$$

These two equations state that if the agent sell (buy) one unit of good  $g$  at price  $\bar{P}_s$  ( $\bar{P}_b$ ), with price measures in the amount of good  $m$ , the agent's utility would be the same as before.

Hu and Wellman [9] has proved that for quasi-concave (such as CES) utility, the agent's reservation buy price is always lower than its reservation sell price. In addition, the agent's buy bid  $P_b$  and sell bid  $P_s$  should satisfy  $P_b \leq \bar{P}_b$  and  $P_s \geq \bar{P}_s$  in order for its utility to increase.

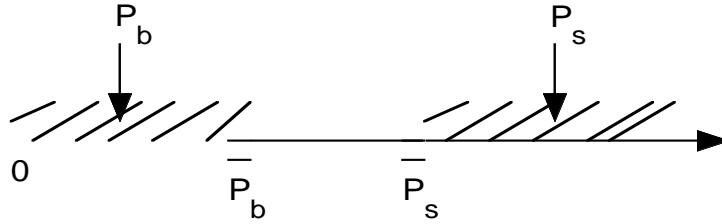


Figure 4: An agent's reservation prices and its actual bids

### 3.5 Three types of agents

We designed three types of agents. One is *competitive* agents who always bid their true reservation prices. The other two are strategic learning agents who

choose their bidding prices based on their possible influence on the market. These include the agents who model the market clearing price and the agents who model the bids of other agents.

Mth-price auctions (of one-period) are incentive compatible for a seller [18], which means the best bid the seller can submit is its true reservation price  $\bar{P}_s$ . The only way for an agent to make profits is through its buy bids.

Both types of strategic agents predict the next period's clearing price, and choose their buy bids as best responses to their predictions. The two types differ in how they use the information in the auction. The price-modeling agent ignores the individual behavior of other bidders by focusing on the aggregate information of the market, the clearing price. The *bidding modeling agent* models the behavior of other bidders. It keeps track of the bids submitted by other agents in previous periods.

A price-modeling agent uses the historical data of clearing prices to predict the next clearing price. It estimates a time series model,

$$P(t) = \alpha P(t-1) + \varepsilon.$$

After predicting the next clearing price  $P(t)$ , the agent then chooses its best response buy bid such that

$$P_b(t) = \min\{\bar{P}_b(t), P(t) - \delta\}, \quad (4)$$

where  $\delta$  is a small constant. Let us see why (4) is a best response. Given the discussion before, the agent's buy bid  $P_b$  should always be below its reservation buy price in order for its utility to increase. If the agent's reservation price is lower than the market clearing price,  $\bar{P}_b(t) \leq P(t)$ , then we have  $P_b \leq \bar{P}_b(t) \leq P(t)$ . Given the property of the Mth-price auction, a buy price that is lower than the clearing price can never be matched. Therefore it does not matter what bid the agent submit. In this case, the simplest bid the agent can submit is its reservation buy price. Thus  $P_b(t) = \bar{P}_b(t)$ . If the agent's reservation price is higher than the clearing price,  $\bar{P}_b(t) > P(t)$ , the agent can be matched as a buyer. The trading price is the clearing price  $P(t)$ . As a buyer, our agent would like the trading price as low as possible. According to the rule of the auction, the clearing price is the Mth highest price among all bids. If our agent submits a bid that is very close to this Mth price, but a little lower, then its bid would become the Mth price (since its original bid is removed, there are M-1 bids left above its buy bid). Thus the agent submits  $P_b(t) = P(t) - \delta$ .

The bidder-modeling agent models the bids of other agents by looking at the history data of those bids, and uses time series techniques to predict the bids in the next time period. For any other agent  $k$ , the bidder-modeling agent predicts agent  $k$ 's bid in the next period,  $P_t^k$ , by

$$P_b^k(t) = \beta P_b^k(t-1) + \varepsilon \quad (5)$$

where  $\beta$  is a parameter estimated from agent  $k$ 's price history.

After forming predictions of other agents' bids, the strategic agent chooses its new bid as a best response to these estimates. Let  $\{\hat{P}_b^1(t), \dots, \hat{P}_b^n(t)\}$  and  $\{\hat{P}_s^1(t), \dots, \hat{P}_s^n(t)\}$  be the strategic agent's projected buy and sell prices of other agents. Let  $P_M$  be the predicted Mth price, and  $P_{M+1}$  be the predicted M+1st price. If  $\bar{P}_b < P_M$ , the agent cannot be matched as a buyer so it does not matter what bid it submits. Since the agent has uncertainty about the actual bids in the market, the best bid it can submit is its reservation price  $\bar{P}_b$ . If  $\bar{P}_b \geq P_M$ , the agent wants to reduce the Mth price so that it can make more profits. The way to do this is to submit a price  $P^b$  that is lower than  $P_M$  but higher than  $P_{M+1}$  so that this price will become the Mth price.

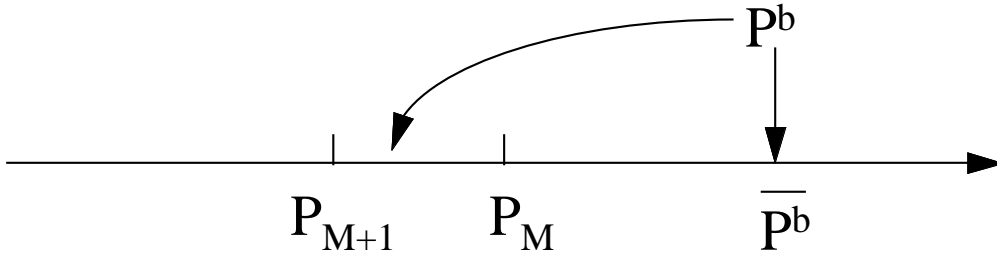


Figure 5: Choose best-response bid

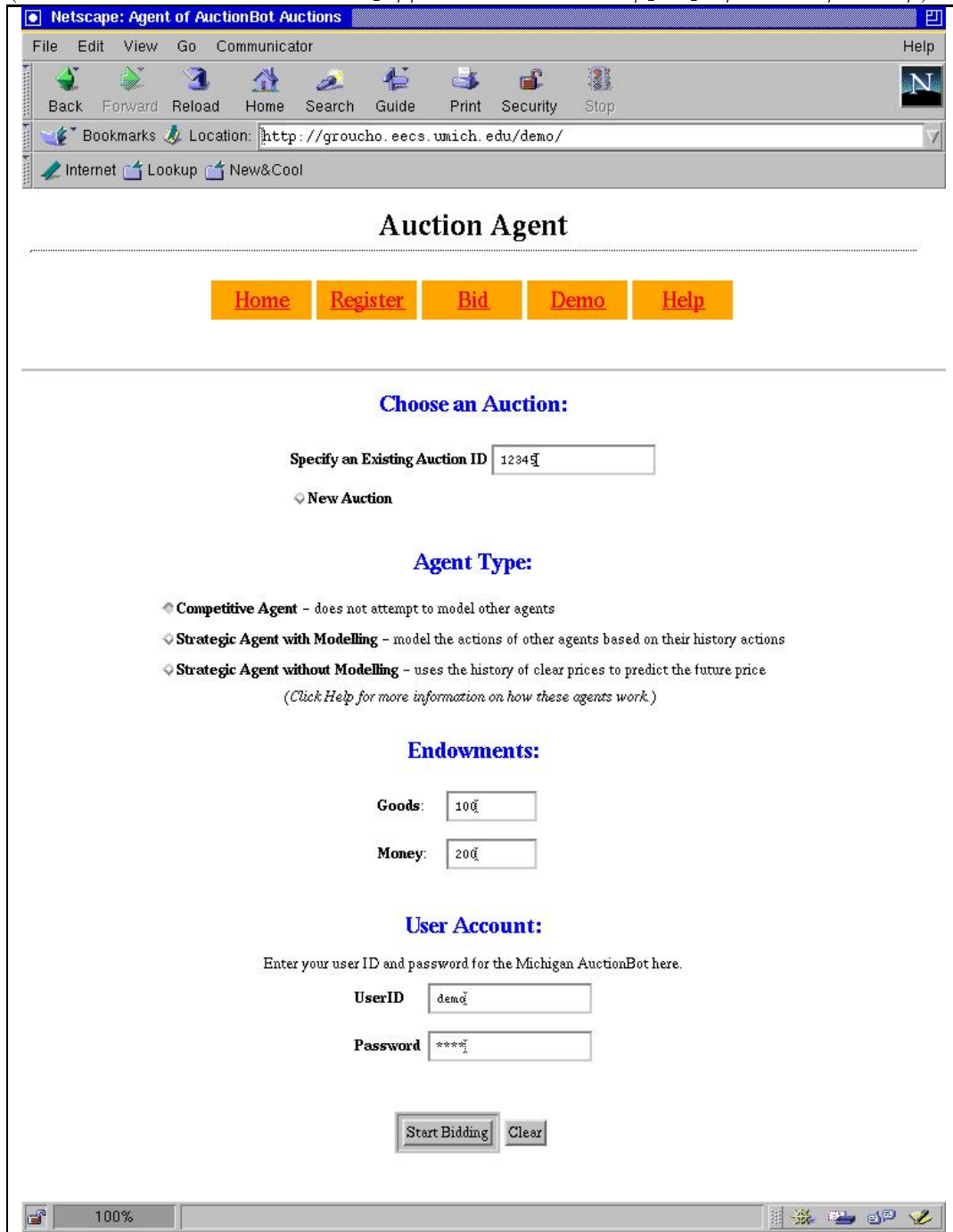
Therefore, the agent's best response buy bid is

$$P_b(t) = \min\{\bar{P}_b(t), P_{M+1} + \epsilon\}, \quad (6)$$

where  $\epsilon$  is a small positive constant representing the minimal bid increment. Note that the above equation automatically satisfies the condition  $P_b(t) \leq \bar{P}_b$  so that the agent's utility will never decrease.

Figure 6:  
Agent server

(The site has now moved to <http://ai.eecs.umich.edu/people/dreeves/demo/>)



## 4 Experiments

There are six agents and two types of goods in our experiments. Each agent starts with a random endowment of both goods. Prices are in units of good 2; thus, all auction activities are for good 1.

Each agent has the same utility function  $U(x) = \left( \sum_j (x_j)^{\frac{1}{2}} \right)^2$ . According to (2) and (3) we have reservation prices as

$$\begin{aligned} \bar{P}_b &= 2(\sqrt{x_g(x_g + 1)} + \sqrt{x_m(x_g + 1)} - \sqrt{x_g x_m} - x_g) - 1, \\ \bar{P}_s &= -2(\sqrt{x_g(x_g - 1)} + \sqrt{x_m(x_g - 1)} - \sqrt{x_g x_m} - x_g) - 1. \end{aligned}$$

We test three types of agents: the competitive agent, price-modeling agent, and bidder-modeling agent. We put these agents in three kinds of environments where all other agents are: (1) competitive agents; (2) price modeling agents; (3) bidder-modeling agents. A competitive agent does not use any information in the auction market. It chooses its action based on its individual optimization problem. A price-modeling agent uses the previous clearing prices to predict the clearing price in the next period and then chooses its best-response bid. A bidder-modeling agent uses other agents' bidding history to predict their next bids and choose its best response bid.

In each of the three environments, we randomly configure the initial endowment of all agents. We compare the performance of the first agent for each type it assumes. Our results are averaged over 6 different sets of initial endowments.

Figure 7 presents results for an environment where all other agents are competitive agents. When Agent 1 chooses the price modeling strategy, at the beginning it performs better than behaving competitively. However, this advantage goes away over time when the price-modeling agent's bid distorts the market clearing price and the auction closes prematurely. Similar results are seen when the agent adopts the bidder-modeling strategy.

In Figure 8, where other agents are price modeling agents, we observe different results. The main difference is that when Agent 1 adopts the bidder-modeling strategy its performance is higher than using other strategies at least for the first 20 rounds. The clearing price strategy outperforms, slightly, the competitive strategy in the earlier rounds but then leads to worse performance than the competitive strategy when it causes the market to close before all trading opportunities have been explored.

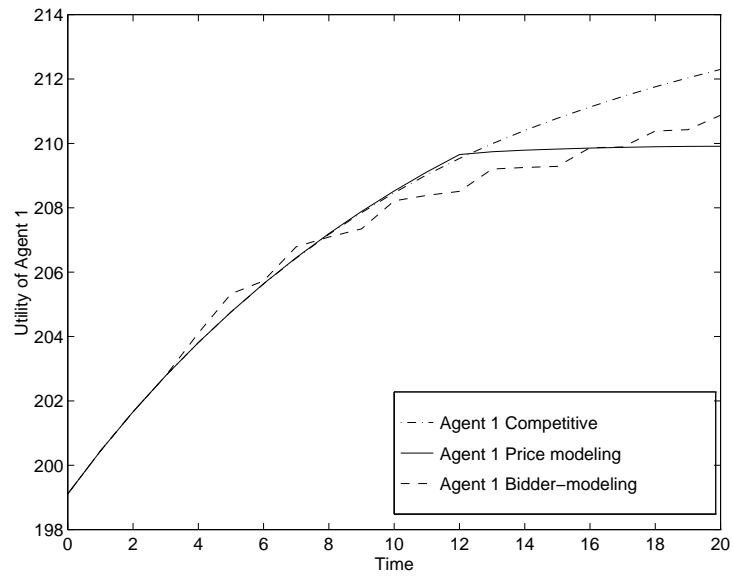


Figure 7: Agent 1's performance when others are competitive agents

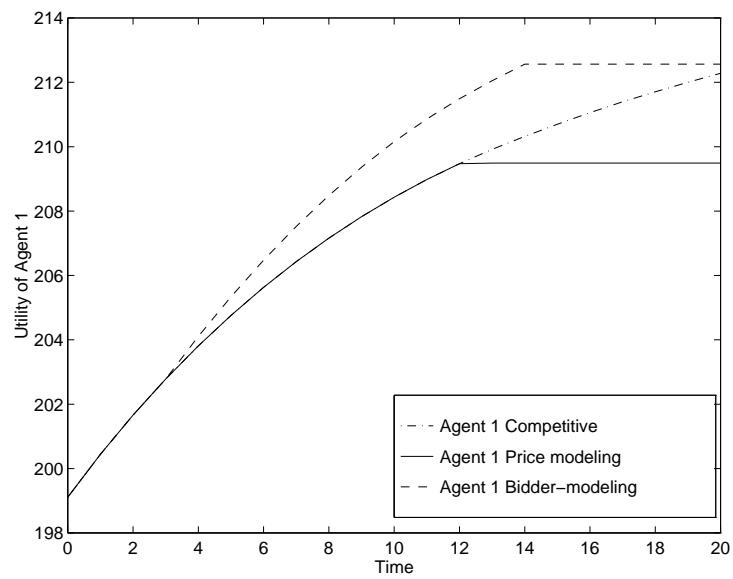


Figure 8: Agent 1's performance when others are price modeling agents

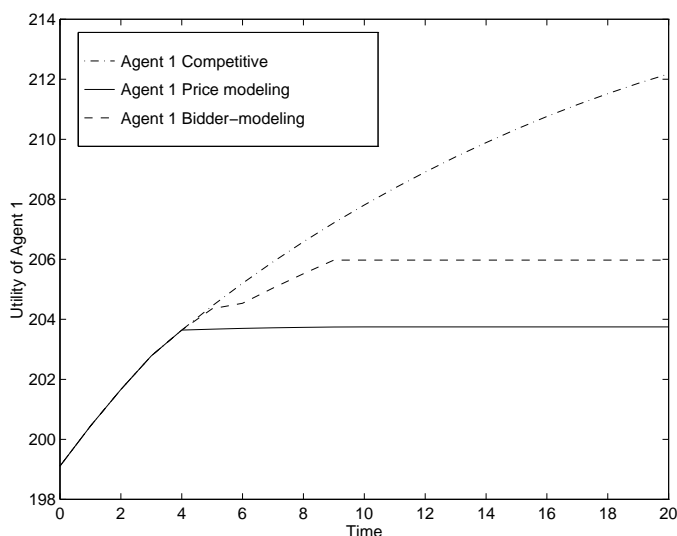


Figure 9: Agent 1’s performance when others are bidders modeling agents

Figure 9 shows the results when others are bidder-modeling agents. In this case both modeling strategies perform worse than the competitive strategy almost all of the time. This is probably because all the agents are trying to model each other rather than bidding truthfully. The system reaches a sub-optimal equilibrium where agents’ behave rationally according to their models of other agents. Such equilibrium was studied in a paper by Wellman and Hu [17].

Our experiments suggest that there is no single best bidding strategy if other bidders in the system behave differently. One solution to this problem is to let the agent identify the types of other agents before it engage in one strategy. The problem with this approach is that there are many possible types of other agents. Another approach is to allow the agent to choose among different strategies at any given time. If one strategy does not work well, the agent then switches to another strategy. It remains a research question as how well this method would work.

## 5 Summary

We have created a configurable agent that can participate in certain auctions hosted on the Michigan AuctionBot using three different bidding strategies.

We perform regression on the bidding histories of other agents and use this to predict a clearing price for the auction. We then make an adjustment to the reservation prices in hopes of getting more profits when the transaction is made. The bidding agents are started with parameters specified by the AuctionBot user such as the type of bidding strategy and the initial endowments.

The agent server was designed for a specific type of auction (Mth-price auctions) but in the future we would like to generalize our agents to participate in other types of auctions as well. Another future enhancement will be to add additional bidding strategies. We will then be collecting detailed performance statistics to determine which strategies perform better under which types of auctions. For example, we can take an evolutionary approach to let different strategies compete with each other and evolve over time. The survival strategy would be the best ones that we are looking for.

## References

- [1] Kurt Bollacker, Steve Lawrence, and C. Lee Giles. Citeseer: An autonomous web agent for automatic retrieval and identification of interesting publications. In *Proceedings of the Second International Conference on Autonomous Agents*, pages 116–120, Minneapolis, May 1998. ACM Press.
- [2] Botspot.com. *Shopping Bots*. <http://botspot.com/s-shop.htm>, 1999.
- [3] Dave Cliff. Evolving parameter sets for adaptive trading agents in continuous double-auction markets. In *Agents-98 workshop on Artificial Societies and Computational Markets*, pages 38–47, Minneapolis, MN, May 1998.
- [4] Robert B. Doorenbos, Oren Etzioni, and Daniel Weld. A scalable comparison shopping agent for the world-wide web. In *Proceedings of the First International Conference on Autonomous Agents*, pages 39–48, 1997.
- [5] David Easley and John Ledyard. Theories of price formation and exchange in double oral auctions. In Friedman and Rust [7], chapter 3, pages 63–98.

- [6] Daniel Friedman. The double auction market institution: A survey. In Friedman and Rust [7], chapter 1, pages 3–26.
- [7] Daniel Friedman and John Rust, editors. *The Double Auction Market*. Addison-Wesley, 1993.
- [8] Dhananjay Gode and Shyam Sunder. Lower bounds for efficiency of surplus extraction in double auctions. In Friedman and Rust [7], chapter 7, pages 199–220.
- [9] Junling Hu and Michael P. Wellman. Online learning about other agents in a dynamic multiagent system. In *Proceedings of the Second International Conference on Autonomous Agents*, pages 239–246, Minneapolis, May 1998. ACM Press.
- [10] R. Preston McAfee and John McMillan. Auctions and bidding. *Journal of Economic Literature*, 25:699–738, 1987.
- [11] Kevin O’Malley and Terence Kelly. An api for internet auctions. *Dr. Dobbs’ Journal*, 289:70–74, September 1998.
- [12] Sunju Park, Edmund H. Durfee, and William P. Birmingham. Emergent properties of a market-based digital library with strategic agents. In *Proceedings of the Third International Conference on Multiagent Systems*, Paris, France, 1998. AAAI Press.
- [13] John Rust, John Miller, and Richard Palmer. Behavior of trading automata in a computerized double auction market. In Friedman and Rust [7], chapter 6, pages 155–198.
- [14] Mark Satterthwaite and Steven Williams. The bayesian theory of the k-double auction. In Friedman and Rust [7], chapter 4, pages 99–124.
- [15] Tripod.com. *Auction agents*. <http://auctions.tripod.com/scripts/AgentMain.asp>, 1999.
- [16] Hal R. Varian. *Microeconomic Analysis*. W. W. Norton & Company, New York, third edition, 1992.
- [17] Michael P. Wellman and Junling Hu. Conjectural equilibrium in multi-agent learning. *Machine Learning*, 33:1–23, 1998.

- [18] Peter R. Wurman, William E. Walsh, and Michael P. Wellman. Flexible double auctions for electronic commerce: Theory and implementation. *Decision Support Systems*, 1998.
- [19] Peter R. Wurman, Michael P. Wellman, and William E. Walsh. The michigan internet auctionbot: A configurable auction server for human and software agents. In *Proceedings of the Second International Conference on Autonomous Agents*, Minneapolis, May 1998. ACM Press.