Agent Service for Online Auctions

Junling Hu and Daniel Reeves and Hock-Shan Wong

Artificial Intelligence Laboratory
University of Michigan
Ann Arbor, MI 48109-2110, USA
{junling,dreeves,hswong}@umich.edu
http://ai.eecs.umich.edu/people/{junling,dreeves,hswong}

Abstract

We have designed configurable agents to represent users in online auctions, specifically the Michigan AuctionBot. The agents can be configured, started, and monitored from a web interface. We implemented three types of 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. When all the agents behave strategically they may reach a sub-optimal equilibrium, in which they receive worse payoffs than behaving competitively.

keywords: *Web agents, internet auctions, online learning, strategic bidding*

Introduction

Intelligent agents for electronic commerce are a popular research topic. There are shopping agents that collect price information for users (Doorenbos, Etzioni, & Weld 1997), and information filtering agents that collect interesting publications (Bollacker, Lawrence, & Giles 1998). We are interested in designing agents for online auctions, where buyers and sellers interact with each other. Such agents can work on the behalf of users since users usually do 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. The interesting research issue for us is how an agent takes advantage of the information available and achieves maximal profit in the transactions. This usually refers to how an agent uses past observations to make predictions and choose its optimal bids. We also address design issues such as how an agent works in a web environment and how it gathers information and makes decisions in real time.

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

Our experiments suggest that an agent's performance in the auction depends not only on its bidding strategy, but also on the bidding strategies of others. A greedy bidding strategy may help the agent to gain in the short run, but may also cause it to lose in the long run.

Design Overview

Michigan AuctionBot

The Michigan AuctionBot (Wurman, Wellman, & Walsh 1998) 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 (McAfee & McMillan 1987). 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 (O'Malley & Kelly 1998) is a client/server communication protocol that is straightforward to implement for client developers in any language on any platform. 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.

Figure 1: Agent server overview

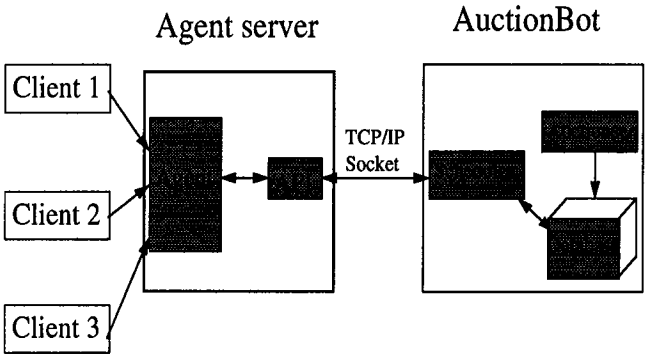
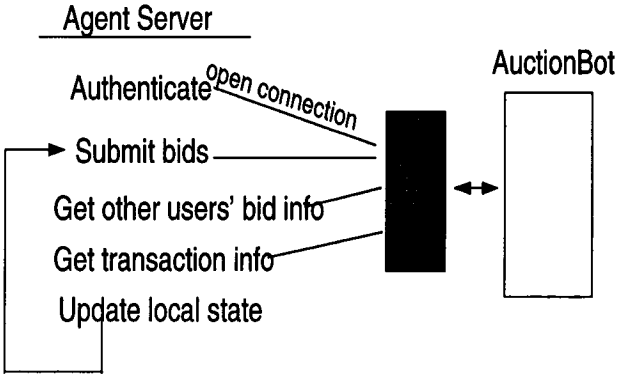


Figure 2: Communication with AuctionBot



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. Once the user ID and password pair is verified, the user's ID, encrypted password, and email address are stored in a protected directory.

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. For security purposes, users have to authenticate themselves every time they request an agent service.

Agents are started through an HTML form and corresponding CGI script which invokes the agent program on the agent server. The agent 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.

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 server or the client stops the connection. The continuous updates allow the user to keep track of the bidding process dynamically.

Communication with AuctionBot

Our agent opens a TCP/IP connection, and then authenticates itself with the AuctionBot. It can then submit bids, get information on other agents' bids, find out about clearing prices, and determine the transactions it has been involved in. The agent can continuously check its bidding status by polling the auction server.

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 between user 782 and 121, one unit was bought for $3.14. The status of zero indicates no error.

The most difficult design issues in the communication with the AuctionBot 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. Figure 2 illustrates this process.

The auction environment

Double auctions

In a *double auction* (Friedman & Rust 1993), 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. This type of auction can be seen in a clearing house. 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 (Friedman 1993) and Internet auctions. The auctions we have designed agents for are synchronized double auctions.

The book edited by Friedman and Rust (Friedman & Rust 1993) collects several studies of double auctions, including both simulations and game-theoretic analyses. Game-theoretic studies (Satterthwaite & Williams 1993) (Friedman 1993) 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 (Easley & Ledyard 1993) try to explain the experimental data generated from human subjects. They assume that each buyer or seller has a reservation price and has a way to recalculate its reservation price after trading. 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.

Gode and Sunder (Gode & Sunder 1993) designed *zero-intelligence* agents who submit random bids within a range such that their utilities never decrease. To improve upon zero-intelligence agents, Cliff (Cliff 1998) 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 (Park, Durfee, & Birmingham 1998) designed an adaptive p-strategy agent to participate in continuous double auctions. Her experiment showed that the p-strategy out-performed other strategies. In the Santa Fe Tournament (Rust, Miller, & Palmer 1993), 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 would jump in and steal the deal. Such an agent is not applicable when there is no negotiation process in the auction.

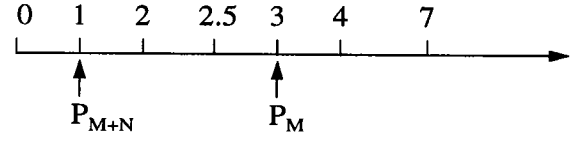
Mth Price Auctions

We are interested in designing agents for Mth-Price Auctions—a subclass of synchronized double auctions (Wurman, Walsh, & Wellman 1998). When there is a single seller ($M = 1$), the Mth-price auction is equivalent to the standard first-price auction, with the provision that the seller is allowed to specify a reservation price.

Consider a set of bids, of which $M (M \geq 1)$ are sell bids, N are buy bids. The Mth-price auction sets the clearing price at the Mth highest among all $M + N$ bids. All buy bids at prices greater than or equal to the clearing price can be matched to all sell bids at prices less than the clearing price. The order of matching is arbitrary.

The Mth-price setting and can be seen more clearly with the example in Figure 3. In the example, the sell bids are {1, 2.5, 3} and the buy bids are {2, 4, 7}. The clearing price is therefore 3, which is the third highest price among all bids. The sellers with bids in {1, 2.5} are then transacted with the buyers having bids in {4, 7}.

Figure 3: An Mth-price auction



Agent design

Modeling the auction

We assume that each agent i has a CES (Constant Elasticity of Substitution) utility function,

$$U(x) = \left(\sum_{g=1}^{m} \alpha_g x_g^{\rho} \right)^{\frac{1}{\rho}}, \quad (1)$$

where $x = (x_1, \ldots, x_m)$ is a vector of goods, the α_g are preference weights, and ρ is the substitution parameter. We choose the CES functional form for its convenience and generality—including quadratic, logarithmic, linear, and many other forms as special cases. In our agent design, we let $\alpha_g = 1$ for all g, and $\rho = \frac{1}{2}$, and so the utility function becomes $U(x) = \left(\sum_g (x_g)^{\frac{1}{2}} \right)^2$.

The reward for agent i at time t is given by

$$\begin{aligned} r_t^i &= U_{t+1}^i - U_t^i \\ &= U(e^i(t+1)) - U(e^i(t)) \end{aligned}$$

In constructing agent strategies, we dictate that they always choose actions leading to nonnegative payoffs. We can characterize this in terms of the agents' *reservation prices* (Varian 1992). The reservation price is defined as the maximum (minimum) price an agent is willing to pay for the good it wants to buy (sell). We can

define agent i's buying and selling reservation prices, $\bar{P}_b$ and $\bar{P}_s$, as the prices such that its utility stays constant when buying or selling one unit of a good.

$$U(e_g - 1, e_m + \bar{P}_s, e_{-g,-m}) = U(e_g, e_m, e_{-g,-m}), \quad (2)$$
$$U(e_g + 1, e_m - \bar{P}_b, e_{-g,-m}) = U(e_g, e_m, e_{-g,-m}). \quad (3)$$

For example, in the case of one good, g, and money, m, the reservation buy price, $\bar{P}_b$, is the price such that the current utility is equal to the utility with one additional good of type g (the one we would be buying) and reduction in money of $\bar{P}_b$ (the price we would pay). In other words, since the utility remains constant, an agent would be indifferent to making such a transaction. At any lower price than the reservation buy price, the agent will increase its utility by transacting.

It is shown in (Hu & Wellman 1998) that for quasi-concave (such as CES) utility, the agent's reservation buy price is always lower than its reservation sell price.

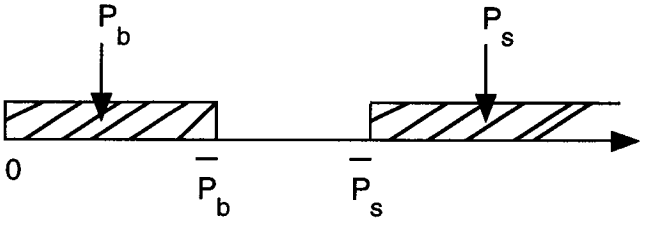


Figure 4: An agent's reservation prices and its actual bids

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. They include price modeling agents and agents who model the bidding of other agents.

Let $\bar{P}^b$ and $\bar{P}^s$ be an agent's reservation buy and sell prices. The best sell bid that an agent can submit is $\bar{P}^s$. This is because Mth-price auctions are incentive compatible for a seller, given that the seller considers only one period's payoff (Wurman, Walsh, & Wellman 1998).

A price modeling agent looks at the history data of clearing prices, and predicts 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, which is the Mth price in the auction, the agent then choose its best response buy bid such that

$$P^b = \min\{\bar{P}^b, P_t - \delta\}, \quad (4)$$

where δ is a predefined constant which reflects the greediness of the agent.

A *bidder-modeling agent* models the actions of other agents by looking at the history data of those actions, and uses time series techniques to predict the actions 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

$$P_t^k = \beta P_{t-1}^k + \varepsilon \quad (5)$$

where β is a parameter estimated from agent k's price history.

After forming predictions of other agents' bids, the strategic agent chooses its new bids as a best response to these estimates.

Let $\{\hat{P}_b^1, \ldots, \hat{P}_b^n\}$ and $\{\hat{P}_s^1, \ldots, \hat{P}_s^n\}$ 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 then 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 profit. 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.

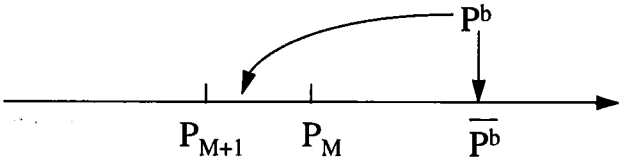


Figure 5: Choose best-response bid

Therefore, the agent's best response buy bid is

$$P^b = \min\{\bar{P}^b, P_{M+1} + \epsilon\}, \quad (6)$$

where ϵ is a small positive constant representing the minimal bid increment. Note that the above equation automatically satisfies the condition $P^b \leq \bar{P}_b$ which means that the agent's utility will never decrease.

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 CES utility function as defined in (1). We let $\alpha_j = 1$ for all j, and $\rho = \frac{1}{2}$, and so the utility function becomes $U(x) = \left(\sum_j (x_j)^{\frac{1}{2}}\right)^2$. According to (2) and (3) we have reservation prices as

$$\bar{P}_b = 2(\sqrt{e_g(e_g+1)} + \sqrt{e_m(e_g+1)} - \sqrt{e_g e_m} - e_g) - 1,$$
$$\bar{P}_s = -2(\sqrt{e_g(e_g-1)} + \sqrt{e_m(e_g-1)} - \sqrt{e_g e_m} - e_g) - 1.$$

We test three types of agents: the competitive agent, price modeling agent, 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 6 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 7, 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.

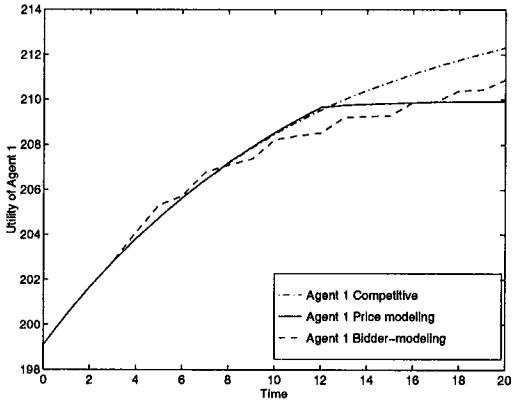


Figure 6: Agent 1's performance when others are competitive agents

Figure 8 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. This leads to some sub-optimal equilibrium, as we have discussed in a previous paper (Wellman & Hu 1998).

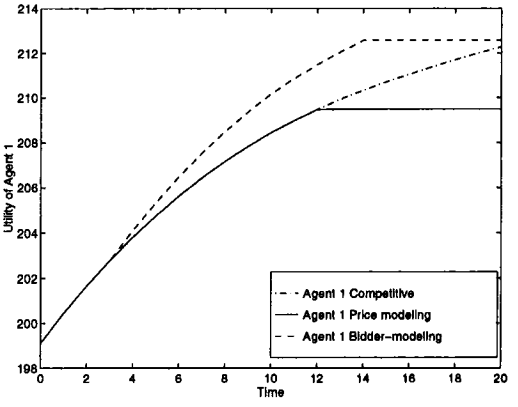


Figure 7: Agent 1's performance when others are price modeling agents

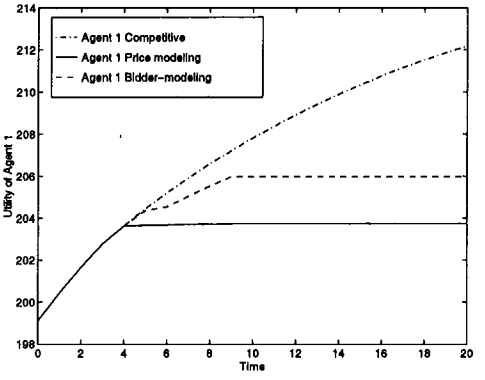


Figure 8: Agent 1's performance when others are bidders modeling agents

Summary

In this project 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 surplus when the transaction is made. The bidding agents are started with parameters specifying the AuctionBot user, the type of bidding strategy, and the initial endowments.

This project was designed for a specific type of auction (synchronous, Mth-price auctions) but in the future we would like to generalize our agents to participate in other types of auctions as well. For example, stock exchanges are asynchronous double auctions. Another future enhancement will be to add additional bidding strategies. We will then be collecting detailed per-

formance statistics to determine which strategies perform better under which types of auctions. We would also like to compare these strategies to human strategies. We will also provide semi-automated bidding services which allow a human to have control over the bidding process.

References

Bollacker, K.; Lawrence, S.; and Giles, C. L. 1998. Citeseer: An autonomous web agent for automatic retrieval and identification of interesting publications. In *Proceedings of the Second International Conference on Autonomous Agents*, 116–120. Minneapolis: ACM Press.

Cliff, D. 1998. Evolving parameter sets for adaptive trading agents in continuous double-auction markets. In *Agents-98 workshop on Artificial Societies and Computational Markets*, 38–47.

Doorenbos, R. B.; Etzioni, O.; and Weld, D. 1997. A scalable comparison shopping agent for the world-wide web. In *Proceedings of the First International Conference on Autonomous Agents*, 39–48.

Easley, D., and Ledyard, J. 1993. Theories of price formation and exchange in double oral auctions. In Friedman and Rust (1993). chapter 3, 63–98.

Friedman, D., and Rust, J., eds. 1993. *The Double Auction Market.* Addison-Wesley.

Friedman, D. 1993. The double auction market institution: A survey. In Friedman and Rust (1993). chapter 1, 3–26.

Gode, D., and Sunder, S. 1993. Lower bounds for efficiency of surplus extraction in double auctions. In Friedman and Rust (1993). chapter 7, 199–220.

Hu, J., and Wellman, M. P. 1998. Online learning about other agents in a dynamic multiagent system. In *Proceedings of the Second International Conference on Autonomous Agents*, 239–246. Minneapolis: ACM Press.

McAfee, R. P., and McMillan, J. 1987. Auctions and bidding. *Journal of Economic Literature* 25:699–738.

O'Malley, K., and Kelly, T. 1998. An api for internet auctions. *Dr. Dobb's Journal* 289:70–74.

Park, S.; Durfee, E. H.; and Birmingham, W. P. 1998. Emergent properties of a market-based digital library with strategic agents. In *Proceedings of the Third International Conference on Multiagent Systems.* Paris, France: AAAI Press.

Rust, J.; Miller, J.; and Palmer, R. 1993. Behavior of trading automata in a computerized double auction market. In Friedman and Rust (1993). chapter 6, 155–198.

Satterthwaite, M., and Williams, S. 1993. The bayesian theory of the k-double auction. In Friedman and Rust (1993). chapter 4, 99–124.

Varian, H. R. 1992. *Microeconomic Analysis.* New York: W. W. Norton & Company, third edition.

Wellman, M. P., and Hu, J. 1998. Conjectural equilibrium in multiagent learning. *Machine Learning* 33:1–23.

Wurman, P. R.; Walsh, W. E.; and Wellman, M. P. 1998. Flexible double auctions for electronic commerce: Theory and implementation. *Decision Support Systems.*

Wurman, P. R.; Wellman, M. P.; and Walsh, W. E. 1998. The michigan internet auctionbot: A configurable auction server for human and software agents. In *Proceedings of the Second International Conference on Autonomous Agents.* Minneapolis: ACM Press.