A Peer-to-Peer Agent Auction

Elth Ogston
Computer Engineering Laboratory, ITS
Delft University of Technology
elth@ce.et.tudelft.nl

Stamatis Vassiliadis
Computer Engineering Laboratory, ITS
Delft University of Technology
stamatis@ce.et.tudelft.nl

ABSTRACT

In this work we examine a peer-to-peer agent continuous double auction. We compare agents trading using peer-to-peer communications with agents using the same trading strategy in an auction that makes use of a centralized auctioneer to disseminate information. We present simulation data for these two auctions running with 2,500 to 160,000 agents. We find that the peer-to-peer auction is able to display price convergence behavior similar to that of the centralized auction. Further, the data shows that the peer-to-peer system has a constant cost in the number of message rounds needed to find the market equilibrium price as the number of traders is increased, in contrast to the linear cost incurred by the central auctioneer. Considering the above message costs, the peer-to-peer system outperformed the simple central auction by at least 100 times in our simulations. We further calculate that for a distributed hierarchical set of auctioneers, for which the message rounds cost of finding equilibrium are reduced to logarithmic in the number of traders, the peer-to-peer system will still produce better performance for systems with more than 5,000 traders.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of Systems

General Terms

Experimentation, Theory

Keywords

auctions, bidding agents, scalability, self-organizing systems, multi-agent simulation

1. INTRODUCTION & RELATED WORK

The auction mechanism, by which self-interested traders are able to settle on a fair price for a commodity, is a key demonstration of the concept of autonomous agents working together without outside control. Moreover the simplicity and robustness of agent auction algorithms make them well suited to a variety of applications; e-commerce naturally, but also more general resource allocation problems [9] [10]. However while an auction demonstrates distributed determination of prices, auctions are most often implemented using a central auctioneer and thus overall are not fully distributed systems. This central auctioneer distributes global information about current prices and deals made among traders. In an agent system running on a single machine or a well connected network, such high quality information is certainly worth the cost of maintaining a central source. However, as agent systems move to run on less reliable networks the communications cost of maintaining a central auctioneer could become prohibitive, limiting the number of auction participants. In this paper we investigate the abilities of a peer-to-peer auction, created by adapting a peer-to-peer matchmaking procedure which we have shown to be effective when agents search for one of a number of randomly placed acceptable partners [4] [5] . We take a simple agent bidding algorithm that has been shown to work well given information about the best bids and offers in an auction and run it with information from only a limited neighborhood of other agents. While the lower quality of this information means that such peer-to-peer traders take more time to find a solution, we find that they never the less are able to converge to the equilibrium price for the market. Moreover, the cost savings in terms of messages to any particular entity in the system are significant. While the number of messages processed by a central auctioneer grows linearly with the number of agents, we find in simulations that the maximum messages to any entity in our peer-to-peer system remains approximately constant both in the message rounds needed to reach equilibrium and the message rounds needed to continue making subsequent deals.

Auction mechanisms are widely studied, from economics where the efficiency of different markets is measured, to game theory where the abilities of market mechanisms to stand up to malicious agents are compared, and finally in computer science where agents are used to buy or sell goods under varying circumstances [3] [7]. Agent market research is based on the core theory that markets produce an efficient allocation of resources even when using agent traders in place of human traders. Economic theory states that there is a computable equilibrium price for a given commodity market based on the best price at which each trader in that market is willing to buy or sell. This equilibrium

AAMAS'02, July 15-19, 2002, Bologna, Italy.

price is the price at which the largest number of trades will be made within the market. Remarkably, human markets settle on this equilibrium price after a time, in spite of the fact that none of the traders in the market know the other traders' best prices. This paper is based on a line of research that explores the question: what is the minimal intelligence required in market participants to create an efficient market? Gode and Sunder first asked this question in [2] where they compared "zero intelligence" agents that bid randomly between lower and upper bounds to experiments with human traders done by Smith [8]. They concluded that the market mechanism itself, and not the intelligence of the traders, was sufficient for the market to settle at equilibrium price. Cliff and Bruten [1] later investigated this question further and showed that such zero intelligence agents only work given certain supply and demand curves. They created "zero intelligence plus" agents that use a simple learning algorithm to make use of past information and showed that it worked in the circumstances where Gode and Sunder's zero intelligence traders failed. Priest and Van Tol [6] later extended this work from an auction mechanism where only one agent bids at a time to a more realistic scenario where all agents bid simultaneously.

This paper introduces peer-to-peer auctioning and compares the central auction and simple learning agents used in [6] with such a peer-to-peer auction using the same agents. This peer-to-peer auction is created by adapting a peer-to-peer matchmaking procedure that we analyzed in [4] and [5]. In that work we were concerned with removing the central brokerage or yellow pages entity from the distributed agents matchmaking problem. We found that this was possible provided that there were a number of acceptable matches for each task and, based on this, that the chance of two randomly picked tasks matching was high enough. A similar problem exists within agent auctions, which in general make use of a central auctioneer. We thus extend the question over the minimum agent abilities for an efficient auction to include the issue of the minimum communication requirements among those agents. In this paper we experimentally measure the number of bidding rounds and number of message rounds required to reach convergence in two forms of auctions, a peer-to-peer auction and a traditional auction centralized around an auctioneer, as a function of the number of trading agents, N. We find that:

- The peer-to-peer auction displays price convergence.
- In bidding rounds, the rate of convergence is independent of N in both auctions and approximately two times faster in the centralized auction.
- The number of messages per bidding round to and from any entity in the peer-to-peer auction is constant, while an auctioneer must handle a number of messages each bidding round that grows linearly with N.
- Considering message round costs, in simulations with 2,500 to 160,000 agents the peer-to-peer auction is at least 100 times more efficient than the centralized auction. Compared with a distributed auctioneer the peer-to-peer system is more efficient for auctions with more than 5000 agents.

In the following sections we will define our peer-to-peer auction and the centralized auction we compare it to, and then measure their performance through simulations. In section 2 we summarize the basic micro-economic model of a market and then describe the exact procedures used in the central and peer-to-peer auction simulations we will be comparing. As we are primarily interested in communications costs we include an analysis of the number of message rounds per bidding round each procedure requires. In section 3 we explain our choice of parameters for comparison and present and analyze our simulation results. Section 4 concludes with some final remarks.

2. EXPERIMENTAL SETUP

Markets consists of a number of *traders*; *sellers* who have an item that they wish to sell and *buyers* who have some money to buy items. In this work we consider a simplified theoretical market with only one commodity good being traded. This means that all items for sale are identical and thus avoids the question of one item being intrinsically more valuable than another. Traders are modelled as having a *reservation price*, a minimum price they are willing to sell items at, or a maximum price they are willing to pay for items. The reservation prices of all traders in the market put together create supply and demand curves, as shown in figure 1. The supply curve shows the number of items sellers are willing to sell at each price, the demand curve analogously shows how many items buyers are willing to buy at each price. The intersection of the two curves is at the *equilibrium price*, P_0, *and equilibrium quantity*, Q_0. This *equilibrium point* is the price and quantity at which the maximum number of items will be exchanged. In theory markets will naturally tend towards trading at this point. If the market price is above equilibrium there will be more sellers competing to trade with fewer buyers thus bringing the price down. Vice versa, if the price is below equilibrium there will be more buyers wanting to buy then items for sale, driving the price up. Thus, a measure of the effectiveness of a market mechanism is how close to equilibrium trades take place. Also, since markets with no prior history will most likely start making trades off equilibrium, a second important measure is how quickly equilibrium is reached. In this work we consider agents trading in a *continuous double auction*. In auctions market prices are determined through bidding rounds in which buyers and sellers shout the current price that they are willing to exchange an item at. If these prices overlap the traders involved make a *deal*, otherwise the traders need to update the price they are willing to shout in the next round. In a continuous double auction both buyers and sellers announce their current asking prices, and shouts remain valid until updated by a new shout in a later round.

In our experiments we create software agents that act as traders in simulated auctions. Each agent represents a single buyer or seller with a fixed reservation price and a single item to buy or sell. Agents update their asking prices according a simple learning algorithm, and once they have made a trade will re-enter the system with probability P_r at each bidding round. We compare these agents trading in two different auction setups, a basic centralized auction and a peer-to-peer auction. Our aim is to determine how effective a peer-to-peer auction might be in reaching equilibrium, and to compare the communication costs of finding trades in the two different setups.

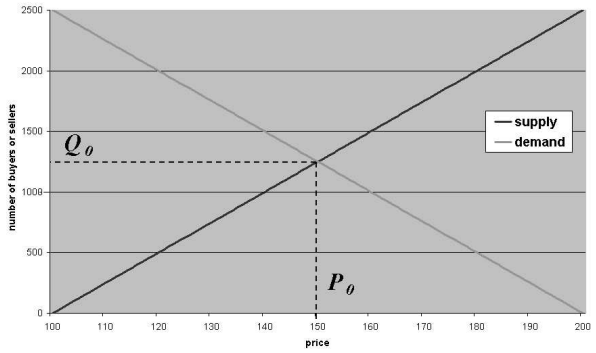

Figure 1: Example supply and demand graph

2.1 Central Auction & Bidding Algorithm

We use a modified version of the centralized auction and the agent bidding update algorithm presented in [6] as our basis point for comparison to our peer-to-peer auction. In this auction bidding takes place in rounds in which each trader that has resources available sends its current asking price to a central auctioneer. This auctioneer determines if there are overlapping bids, and if so pairs up trading buyers and sellers. The buyer with the highest bid is paired to the seller with the lowest offer, second highest bid to second lowest offer and so on until there is no more overlap. Trades are made at the average of the paired buyer's and seller's asking prices. The auctioneer then broadcasts the best bid and offer to all the agents in the market. This information is used by the agents to determine what price to shout in the next round. Agents who have traded have their good or money replaced with constant probability P_r each round after having made a trade.

If you consider each agent and the auctioneer to be separate entities, each able to send or receive one message at a time, the serial communication costs of this centralized procedure are high. During a bidding round traders first send bids to the auctioneer. These must be received and processed, giving a cost of up to N message rounds, where N is the number of traders. The auctioneer must then send a message to each trader in turn with the current market information and that trader's deal status. As traders continue to update their prices, even when they have no resources, this cost is always N messages. Thus we have a total cost of up to $2N$ message rounds per bidding round. By distributing the auctioneer this can be reduced to $O(\ln N)$, however as this involves other complications we will leave this analysis until section 3.3 and for now consider the simpler single auctioneer.

The bidding update algorithm is a heuristic update rule, with a simple learning element, designed to demonstrate that even very simple agents can be used to create effective markets. Agents are buyers or sellers, each with a single fixed parameter, their reservation price R_0, and a variable current asking price $p(t)$, where $p(0)$ is a random value between the reservation price and a minimum for buyers, or a maximum value for sellers. Each round agents are given the minimum selling price shouted, S_{min}, and maximum buying price shouted, B_{max}. They use this to determine a target price, $\tau(t)$, to update $p(t)$ towards using the following heuristic:

$$\tau(t) = \begin{cases} \begin{cases} B_{max}+\delta, & \text{if } S_{min} > B_{max} \\ S_{min}-\delta, & \text{if } S_{min} \le B_{max} \end{cases} & \text{for buyers} \\ \begin{cases} S_{min}-\delta, & \text{if } S_{min} > B_{max} \\ B_{max}+\delta, & \text{if } S_{min} \le B_{max} \end{cases} & \text{for sellers,} \end{cases}$$

where $\delta = r_1 p(t) + r_2$ and r_1 and r_2 are uniformly distributed random variables taken independently from the intervals $(0, R_1]$ and $(0, R_2]$ respectively. The parameters R_1 and R_2 are used to define a small amount of random variation designed to model differences among traders. Agents use the target price to determine their next shout price following the learning rule:

$$p(t+1) = \begin{cases} \max\{p(t)+\Gamma(t+1), R_0\}, & \text{for sellers} \\ \min\{p(t)+\Gamma(t+1), R_0\}, & \text{for buyers,} \end{cases}$$

where

$$\begin{aligned} \Gamma(t+1) &= \gamma\Gamma(t) + (1-\gamma)\beta(\tau(t)-p(t)), \text{ and} \\ \Gamma(0) &= 0. \end{aligned}$$

This uses two variables, the learning rate, $0 \le \beta \le 1$ which determines to what extent the new price is based on the target price for this round, and the momentum, $0 \le \gamma \le 1$, which determines to what extent the current change in price should be the same as that in the last round.

2.2 Peer-to-Peer Auction

In [4] and [5] we consider agents connected by a random peer-to-peer communications network which searched for partners for tasks by forming small groups that locally modified their network connections. The aim of this work was to discover under what conditions matchmaking could occur in a multi-agent system without a pre-defined directory structure. We found that given random connections and a high enough percentage of redundant tasks, matches could indeed be found in such a system. In this work we reason that finding trading partners in a market is similar to matchmaking and thus adapt the distributed matchmaking procedure to create a distributed auction. In this section we will describe in detail the resulting peer-to-peer agent auction procedure. Figure 2 diagrams a possible configuration of some agents within this system to help illustrate the terms we introduce.

As in the centralized auction, our peer-to-peer auction is made up of agents representing individual traders each with a single good to buy or sell, and a fixed reservation price. In the initial setup these traders are each paired with a *neighbor* agent, chosen at random among all the other agents. Additionally seed *basic clusters*, sets of k cooperating agents, are created, also at random. This setup provides agents with an initial neighborhood in which to search for trading partners. As the auction proceeds the basic clusters combine to form larger *compound clusters*, expanding their agents' search spaces. The randomness of the initial setup is assumed to come from, for instance, the placement of agents on computers in a network, and from the assumption that reservation prices are based on so many factors that they are likely to have a random distribution.

From the initial setup bidding rounds, or turns, proceed as follows. Agents first exchange their current asking price with their neighbor. If the asking prices overlap a trade is

Figure 2: A snapshot of peer-to-peer trading agents

made at the average of the two prices. Agents then update their asking price according to the update algorithm used in the centralized auction, described in section 2.1. However, in place of using the system best bid and offer, as this is no longer provided, agents simply use the last seen bid and offer, or their own price if it is better than one of these. We experimented with agents remembering the last m bids and offers seen, however we found that this merely slowed convergence, the memory in the bidding algorithm appeared to be enough to produce satisfactory convergence.

Agents trade independently, however they cooperate in groups, called clusters, in their search for trading partners. Within each cluster a single agent is assigned as the *cluster center*. This cluster center keeps a map of the agents in the cluster; including a list of *unmatched* agents, those who have an item to buy or sell, *matched* agents, those who have traded, and *linked* agents, those who have traded and formed a connection joining two clusters. Each turn, after updates have been made, the cluster center tells each unmatched agent the address of another unmatched agent to pass its current unwanted neighbor on to, thus shuffling the agents' neighbors. This gives each unmatched agent a new partner to exchange offers with on the following round. In this way agents find themselves bidding against all of the cluster's neighbors.

We further provide a grouping mechanism by which the bidding neighborhoods of agents are enlarged. When two agents make a trade they have a chance of forming a link that joins their two clusters. When this happens the clusters involved are combined into a single larger cluster, with the cluster center of the larger initial cluster becoming the new cluster center. Thus for the next bidding round the neighbors of all the unmatched agents in this larger cluster are shuffled. To prevent clusters from becoming too large and ungainly we give them a maximum size restriction of s basic clusters. Once clusters have reached this size trades continue to be made but no new links are formed. Experiments in [5] showed that without this size restriction all the basic clusters in the system would eventually group into a single large cluster, making the cluster center equivalent to the central auctioneer that we wish to avoid.

To limit the number of operations a cluster is involved in we stipulate that each cluster can only form one new link per turn. This requires coordination among clusters to prevent a cluster from receiving two or more simultaneous requests to form links with others. We achieve this by having each cluster designate one of its matched (but unlinked) agents, chosen at random each turn, as being able to form a link. When two neighboring matched agents are both designated by their respective clusters simultaneously the link is agreed and they inform their clusters that they should merge.

Agents that have traded are considered to be out of the trading game for some amount of time, as they no longer have money or an item to sell. Thus, once formed, links persist for some amount of time. However, links are not permanent. Each bidding round, agents that have traded have their item or money replaced with probability P_r. For simplicity we assume that both the buyer and seller agent involved in a trade receive new resources at the same time. Since these agents are now ready to trade again any link between them is broken. The cluster center updates its cluster map and if this was the last link between two parts of the cluster a new center is assigned and the cluster is split. Thus over time clusters are created and broken down again and their set of neighbors changes.

In the experiments in this paper we are interested in measuring the communications cost of this procedure. Thus we need to break down the bidding rounds just described into the sequence of messages agents must process. The procedure is thought of as occurring among agents residing in many machines on a network, and thus cluster and agent computations can take place in parallel. However it is difficult to simulate all of the complications of a truly asynchronous network. For this reason we make some simplifying assumptions to keep our experiments manageable. We assume messages always arrive in zero time, thus avoid the issue of lost or delayed messages and allowing us to assume that if a message has been sent it has been acted upon. In addition we define a turn in our simulation as a period in which all clusters perform the sequence described above once, simultaneously. This gives us an easy time period for taking measurements and making comparisons to the central auction. It however avoids issues of clusters moving at different speeds; instead we simulate all clusters moving at the speed of the slowest cluster. Thus we do not consider that smaller clusters actually could make more moves in this

time period, or clusters having to wait for replies from slower ones.

Given these assumptions, a bidding round for a cluster can be split into four phases; shuffle, exchange, reply and update. In the following paragraphs we shall describe these and calculate the number of messages rounds that each phase must involve. When counting the message costs for these phases we assume that all agents can send messages in parallel, but that individual agents must process messages on at a time. To maximize the message parallelism for operations that involve communications from a center to all cluster agents or vice versa we use the basic cluster centers as middlemen between agents and their cluster centers.

In the shuffle phase the cluster center sends a message to each of its unmatched agents informing them which other unmatched agent in the cluster to pass its unwanted neighbor on to. It also selects a matched but unlinked agent at random and sends it the message that it is designated as being able to form a link this turn. At most this phase involves a message from the cluster center agent to each of the other agents in the cluster. This will cost $s-1$ messages to each of the other basic cluster centers, informing them of the messages to pass on to their agents, and then $k-1$ messages within each of the basic clusters as the basic cluster centers inform each of the other basic cluster agents. Thus the total maximum message count is $s+k-2$.

In the exchange phase each unmatched agent in a cluster simultaneously first disconnects from its neighbor by sending a disconnect messages, then sends the address of this old neighbor to the cluster agent designated by the cluster center in the shuffle phase. This receiving agent then sends a connect message to what is now its new neighbor and the new neighbor sends an acknowledgement back. It is however possible for two clusters to be moving at the same time, and thus an agent can receive a disconnect message from its neighbor before this phase begins. In this case the agent has no neighbor's address to pass on, instead it must wait while its old neighbor passes on its address. It will then receive a connect message which it can pass on to its receiving agent and the receiving agent can send the acknowledgement to the new neighbor. For our simple trading application's purposes the connect message can also include a bid from the sending agent and the acknowledgement can include a return bid. From these bids each agent can separately determine if a trade has been made. As in the centralized market we have trades take place at the average of the two bids involved. Also during this phase any matched agent that has been designated as being able to form a link can send a message with this fact to its neighbor. If it receives such a message as well each agent knows that a link should be created. The maximum number of sequential messages in this phase is 4.

In the reply phase each agent that has changed state must inform the cluster center of this fact. Thus agents that have traded, agents that have formed a link, and agents that have received new resources must all send a message to the cluster center. This can be done by each agent sending a message to its basic cluster center, costing up to $k-1$ messages simultaneously in each basic cluster and then the basic clusters combining their information into a message to the cluster center, costing a further $s-1$ messages. The maximum total message count for this phase is thus $k+s-2$.

Finally, in the update phase the cluster center must consider all these reply messages and rearrange the cluster accordingly. Agents that have trades are moved to the matched list. We then form a new link, if one was agreed. The link setup messages can contain the size of the cluster, and thus we can specify that the largest cluster will create the link. For two clusters of the same size the cluster of the selling agent creates the link. To do this, the smaller cluster's center sends a message to the larger cluster's center containing its map. The larger cluster's center can then calculate the map for the new combine cluster, and send a message to each of its new basic clusters informing them of their new cluster center. There will be at most $\lceil s/2 \rceil$ of these, minus one for the previous cluster center who already knows of the changes. In a second stage, agents that have received new resources must be moved back into the unmatched list. If any agents who were involved in a link have received a new resource, that link must be broken. The cluster center can again calculate the new map, and if the cluster is split by breaking these links can inform the involved basic clusters, designating one per new cluster as the new cluster center and giving it its new map. In the worst case this will leave the cluster center in a lone basic cluster, giving a maximum message cost of $k-1$. Finally, when recalculating the map any matched agents that are both members of the same resulting final cluster can be updated to linked agents without extra cost. The total maximum cost of this phase is $\lceil s/2 \rceil + k - 1$. Our maximum message round cost for a bidding round is: $\lceil 5s/2 \rceil + 3k - 1$.

3. EXPERIMENTAL RESULTS

In this section we present results from simulations of the peer-to-peer and centralized auctions. We consider for both the number of bidding rounds and number of message rounds they require to settle at market equilibrium, and later to make subsequent deals. First however we look more closely at details of the peer-to-peer auction to determine that it does in fact settle to an equilibrium involving all the agents in the system. We then discuss our choice of the parameters to use when comparing the two systems. Finally we present this comparison, analyzing how the two systems behave as the number of agents is varied.

3.1 Peer-to-peer Basics

Before comparing our peer-to-peer procedure to the centralized one we must establish that it produces correct market behavior. As described in section 1, market trade prices should converge over time to a theoretical equilibrium price P_0. In all of our experiments we assign each agent a random reservation price between 100 and 200, creating approximately the supply and demand curves shown in Figure 1. Agents are also given initial shouts at random, for sellers from the interval $[R_0, 299]$, and for buyers from the interval $[1, R_0]$, where R_0 is the reservation price for that particular agent. We create approximately equal numbers of each agent type by assigning each agent to be a buyer or seller at random with a 50% chance of each. P_r, the new resource arrival rate is set to 0.1. Figure 3 shows the series of trade prices produced in a peer-to-peer trail with 2,500 agents. We see that these indeed start out widely scattered and over time converge within 1.3 units of the equilibrium price of 149.6. Smith in [8] introduced a way of appraising how close a set of n trade prices p_i are to equilibrium,

$\alpha = 100 * \left(\sqrt{\left(\sum_{i=1}^{n} (p_i - P_0)^2 \right) / n} \right) / P_0$, a measure of the standard deviation of trade prices from the equilibrium trade price. Graphing α over time gives us a quantification of how quickly an auction converges to equilibrium, and how closely it matches that equilibrium after convergence. In the following experiments we calculate α for the trades that have occurred every round in our peer-to-peer market and every two rounds in the centralized market. In Smith's experiments with humans α settled at between 0.6 and 13.2, Thus we consider an alpha value of 1 or 2 to be reasonably low. In Priest and Van Tol's experiments in [6] agents were shown to converge to an alpha of about 1. We will consider two characteristics of the alpha curves when comparing the two auction types, first the number of bidding rounds it takes to reach a particular alpha value, and second the average value of alpha that the market eventually stabilizes on.

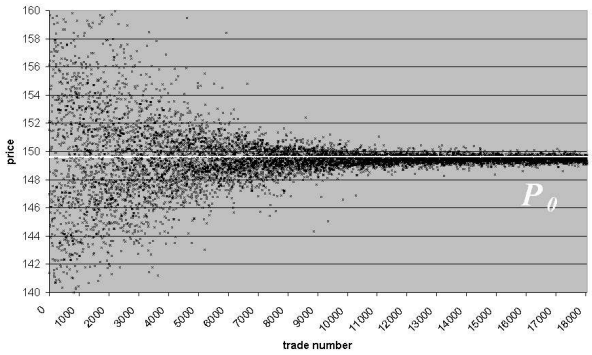

Figure 3: Sample run trade prices

Figure 4 shows the α curve produced by the trade price data in Figure 3. We see that in this run the auction converges to an alpha of 1 in 170 bidding rounds and it eventually converges to an average alpha of .17. We believe this shows acceptable market convergence behavior for the peer-to-peer agents. We must, however, further consider to what extent each agent is participating in the market. For instance, if only half the agents who can trade are trading the random distribution of reservation prices would still result in the same equilibrium price. Figure 5 shows for a longer run the number of trades made by each seller agent, with agents ordered by reservation price. We see that all of the sellers with a reservation price below P_0 do trade. Each make between 30 and 58 trades, independent of how far their reservation price is from P_0. More importantly we see that all traders that should be able to do trade, while there is a steep reduction in the number of trades for sellers with a reservation price near P_0, dropping to no trades being made by most agents with R_0 above P_0. Data for the buyers looks similar to that for the sellers.

We also need to inspect the distribution of agents' trading partners. It is possible that agents repeatedly trade with the same partners, so that the system forms fixed clusters rather than changing organization over time. Figures 6 and 7 show trade partner data for a sample buyer in a trail in which it made 2,706 trades. Figure 6 shows how many trades it made with each of the sellers, ordered by the sellers' reservation prices. We see that the sample buyer traded with almost all of the sellers with a reservation price below P_0. Figure 7 shows the distribution of the number of times the buyer traded with each partner for sellers with a reservation price below P_0. We see that only slightly more than 2% of possible partners were not traded with. For a completely even distribution the buyer agent should have traded with each seller 4.26 times. Indeed we see that the distribution centers near this point. Looking at other agents in the system we find similar trade partner distributions.

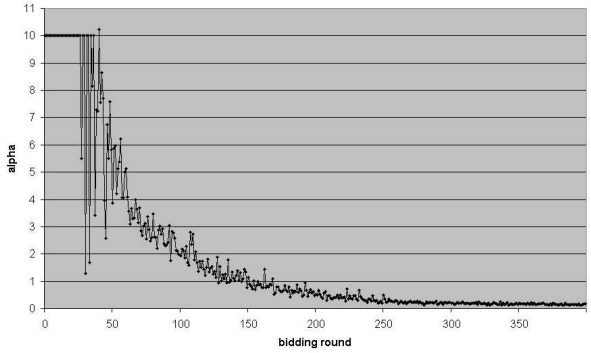

Figure 4: Sample alpha curve

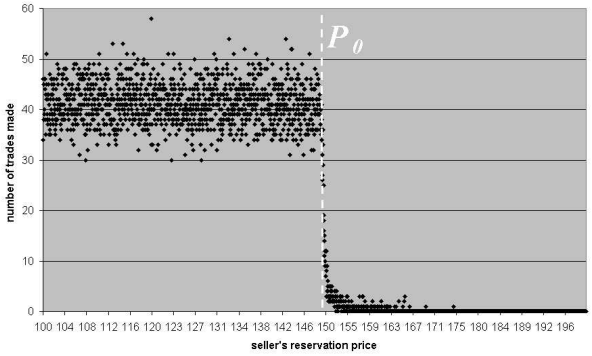

Figure 5: Number of trades per seller

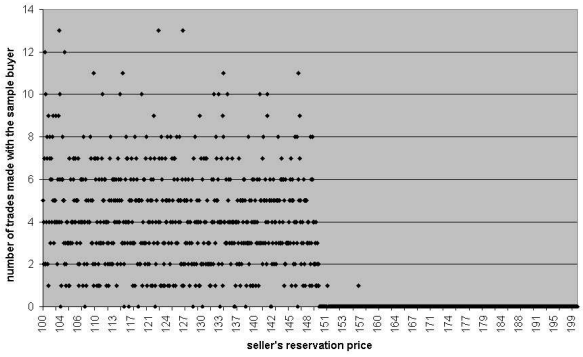

Figure 6: Trades between a buyer and each seller.

3.2 Parameter Choice

Having established that the peer-to-peer auction produces acceptable price convergence and that all agents participate roughly equally, we can now compare the peer-to-peer auction's performance to that of the centralized auction. In doing so however we must consider that performance in both auctions depends upon the parameters used. In each we must consider parameters for the agents' price learning algorithm and for the peer-to-peer auction we must look at the cluster size limitations as well. Indeed, changes in the parameters lead to tradeoffs among different performance characteristics. Moreover, each market performs well with different parameter sets for the bidding update procedure.

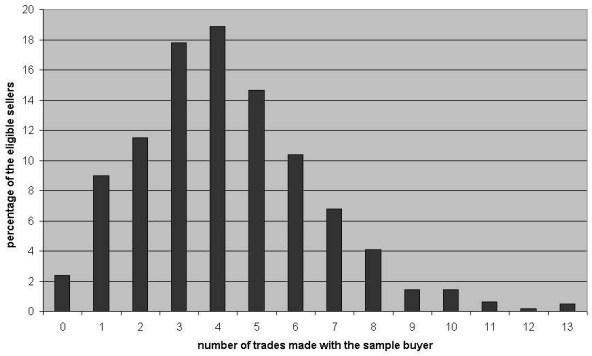

Figure 7: Distribution of buyer's trades

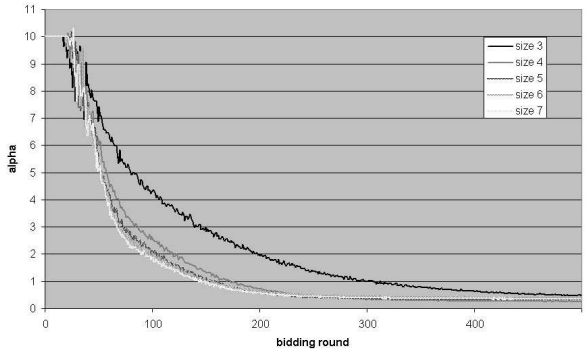

Figure 8: Average α curves, varying cluster sizes

Thus in the following section we discuss our choice of the parameters to use in our comparison experiments.

Agents' reservation prices, buyer or seller status, and initial shouts are assigned at random in both auctions, as described in section 3.1. Both auctions have a further parameter, P_r, the rate at which new resources arrive. Further, the peer-to-peer auction also contains s, the maximum cluster size and k,the number of agents in a basic cluster. For both auctions the new resource arrival rate, P_r, is set at 0.1, a value taken from experiments in [4] which was shown to work well for the matchmaking procedure. The number of agents in a basic cluster, k, and the maximum cluster size, s, determine how many message rounds are needed in a cluster each bidding round. For this reason we would like to keep them both small. However, experiments in [5] and [4] show that s and k also affect how quickly clusters form since they determine how many potential partners each agent has. The more quickly clusters form, the more partners agents see, and the faster trade price convergence occurs. Furthermore, if s or k are too small compared to the chance of two agents being able to trade clusters won't form at all. For operations that involve messages between a cluster center and cluster agents it is most efficient to set s and k to the same value. Thus we ran experiments with the (s, k) pairs, (3,3), (4,4), (5,5), (6,6) and (7,7). Figure 8 shows the α curves for the average of 25 trials at each of these sizes. We see that the convergence rate is improved by increasing (s, k), but that that this improvement lessens as the values increase. Examining the number of messages rounds until α reaches 1 we find that at the point $s = 5$, $k = 5$ the increase in convergence rate no longer makes up for the greater number of message rounds per bidding round. Thus in the following experiments we use the values $s = 5$, $k = 5$.

For both auctions the agent bid update algorithm has the following parameters, all of which affect the rate at which an agent changes its asking price: Momentum (γ), the extent to which price changes are based on previous price changes, learning rate (β), how quickly an agent moves towards its current target price, R_1, the upper limit of a fudge based on current price and R_2, upper limit of a fixed fudge to ensure that prices always move a bit. Of these we found that the momentum made the most difference to performance. Momentum determines how willing an agent is to change its price more or less than it did the turn before. With dependable information, as in the central auction, momentum can be low, as the agents can be fairly sure that their target price represents the market as a whole. On the other hand in the peer-to-peer model agents are given very low quality information and thus need a high momentum, meaning that they base their price changes more on the history of target prices than on just the current one. Based on this analysis, and some exploration done by hand, we choose values for the momentum in the two auctions; $\gamma = 0.05$ for the centralized auction, and $\gamma = 0.9$ for the peer-to-peer auction. From this point we used more hand tuning and a genetic algorithm search to determine good values for the other bid update parameters. Our aim in determining these parameters was to obtain the fastest possible dependable convergence while keeping the α values after convergence below 1. These two goals are often at odds, agents that change prices quickly will converge more quickly, but are also more likely to jump to prices further from equilibrium once convergence has occurred. For instance, increasing the learning rate, which determines how much agents are willing to jump each turn towards their opponent's price, decreases the time to convergence. However, if the learning rate is set too high, values for α vary greatly after convergence. Similarly with R_1 and R_2, which create some randomness that is used to keep the market moving. High values for R_1 and R_2 lead to faster convergence, but again lowering their values creates a more stable end solution. We confirmed that the values used in [6] produce good behavior in the centralized auction, and thus use the same values in our comparison experiments: $\gamma = 0.05$, $\beta = 0.3$, $R_1 = 0.2$, $R_2 = 0.2$. We found that the high momentum used in the peer-to-peer auction resulted in a fairly unstable alpha after convergence, and to counteract this effect we lowered values of the learning rate, R_1 and R_2. The resulting parameters for the peer-to-peer auction were thus: $\gamma = 0.9$, $\beta = 0.25$, $R_1 = 0.001$, $R_2 = 0.02$.

3.3 Comparison of the Two Auctions

We now investigate how the behavior in the peer-to-peer and centralized auctions changes as we increase the number of agents participating. For each market type we ran 100 trials with 2,500, 5,000, 10,000, 20,000, 40,000, 80,000 and 160,000 agents using the parameters chosen in section 3.2. A summary of the data from these trials is given in Table 1. In the following sections we consider more closely the number of bidding rounds it took the auctions to reach equilibrium, the number of message rounds this involved, and after equilibrium is established, how many message rounds are needed to make each subsequent deal.

Figure 9 shows the average α curves for the centralized and peer-to-peer auctions with 2,500, 10,000 and 40,000 agents. We find that the centralized auction converges roughly two times faster. We also find that as the number of agents is increased the rate of convergence in both markets remains approximately the same. The last three columns of Table 1

CENTRALIZED SYSTEM DATA												
	bid. rounds to alpha=2.12			message rounds to alpha=2.12			bid. rounds/agent deal	mes. rounds/agent deal	end alpha value		
number of agents	min	avg	max	min	avg	max	avg	avg	min	avg	max
2,500	36	53.72	88	167326	246513.18	399055	19.53	88343	0.05	0.40	2.02
5,000	38	54.38	76	351416	498646.49	687750	19.65	177974	0.03	0.34	1.16
10,000	42	55.00	66	776057	1009479.68	1205547	19.87	360268	0.03	0.31	1.18
20,000	46	55.74	68	1696591	2046734.75	2482085	20.02	726712	0.02	0.33	0.95
40,000	48	55.12	70	3537399	4048187.55	5107476	20.15	1464140	0.01	0.29	0.72
80,000	50	55.16	64	7360688	8104125.25	9389452	20.27	2947772	0.02	0.30	0.63
160,000	52	55.77	60	15305138	16404413.02	17636285	20.37	5929863	0.05	0.29	0.53
PEER-TO-PEER SYSTEM DATA											
2,500	102	127.16	224	1529	2038.64	3983	22.30	447.01	0.05	0.39	1.62
5,000	97	117.77	147	1486	1914.06	2516	22.30	461.31	0.07	0.31	1.02
10,000	100	109.08	130	1565	1788.13	2226	22.29	473.60	0.08	0.22	0.62
20,000	95	105.02	120	1529	1747.39	2056	22.34	485.12	0.08	0.20	0.52
40,000	95	101.60	113	1565	1711.23	1968	22.29	494.10	0.09	0.15	0.40
80,000	96	100.75	109	1628	1729.18	1922	22.64	509.66	0.11	0.17	0.37
160,000	97	101.17	105	1679	1770.83	1868	22.62	517.15	0.11	0.16	0.25
HIERARCHICAL CENTRALIZED SYSTEM ESTIMATIONS											
2,500				1531.29	2285.03	3743.16		830.67			
5,000				1759.56	2518.03	3519.12		910.04			
10,000				2103.05	2753.99	3304.79		994.77			
20,000				2476.68	3001.09	3661.18		1077.67			
40,000				2765.25	3175.42	4032.65		1160.79			
80,000				3068.88	3385.59	3928.17		1244.01			
160,000				3387.59	3633.14	3908.76		1327.23			

Table 1: experimental data summary

show minimum, average and maximum α values after convergence, averaged over the last 100 bidding rounds of each trial. We find that trials with more agents are more exact; as more agents are added the spread of end alpha values, and more importantly the maximum end alpha value decreases. This causes complications when picking an α value at which to measure the time to convergence. Over all the trials the highest α after convergence was 2.12 and thus we choose to measure convergence as the number of rounds taken to reach an α of 2.12. This is high for the trials with many agents but makes the points with fewer agents easier to read. Using an α of 1 as we did earlier gives similar results but makes interpreting points with fewer agents less clear-cut. From Table 1 columns 2 to 4 we can see that the number of bidding rounds to α=2.12 remains almost constant as the number of agents increases at an average of about 100 bidding rounds in the peer-to-peer auction and about 55 bidding rounds in the centralized auction. In the peer-to-peer case the average number of bidding rounds decreases a little as the maximum value drops towards the minimum.

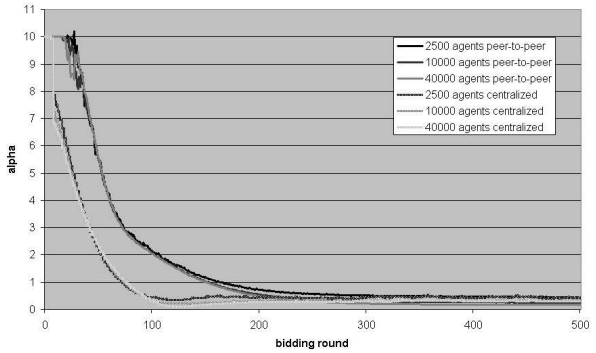

Figure 9: Average α curves, varying system sizes

For an auction running over a network it is likely that communications costs will make a large contribution to the actual time taken by the bidding rounds graphed in figure 9. Thus we also consider the number of message rounds that occur within the auctions up to our convergence point. For the centralized auction we count the message rounds required by the auctioneer each trading round as discussed in section 2.1. For the peer-to-peer auction we count the maximum number of message rounds required by any cluster in a trading round, as discussed in section 2.2. We sum these for the bidding rounds up to and including the first trading round where alpha is below 2.12, giving the results in columns 5 to 7 of Table 1. We see that the number of message rounds required to reach equilibrium in the centralized auction increases linearly with the number of agents, as the auctioneer has to deal with increasingly more bids. In the peer-to-peer auction, on the other hand, it remains pretty much constant since the maximum size of the clusters does not change. For the large system sizes tested in our experiments the peer-to-peer auction always outperformed the centralized auction on this measure. Extrapolating from the data the two systems should be equal on this measure at around 165 agents.

Finally, Table 1 columns 8 and 9 also presents data on the time agents must wait between each deal they make. This becomes more important than the cost of converging in long running auctions where the supply and demand curves do not change quickly. The data shown was determined by counting the message rounds and deals during the last 100 bidding rounds, averaged over 100 trials. We estimated that half the agents in each trial, those with a reservation price that allowed them to trade, were making these deals. Column 8 shows that for both the peer-to-peer and centralized auction the average number of bidding rounds it takes an agent to make a deal remains approximately constant. Column 9 however shows that when considering the message rounds the time cost of trades in the centralized auction increases linearly. For the peer-to-peer auction we expect a constant cost, however we also see a small increase. There is a maximum of 26 message rounds per bidding rounds given our values of $s = 5$, $k = 5$. This is a maximum value, however, most cluster's bidding rounds take less time. The increase in message rounds in column 9 occurs because the average number of message rounds per trading round increases from about 20 to 23 as the number of clusters is increased. However, provided that the number of bidding rounds per agent deal remains constant at about 23 the number of message rounds per agent deal will be capped at 598.

Our experiments considered a simple centralized auctioneer, however, in a large system the benefits of distributing this auctioneer outweigh the costs. Such a distributed central auctioneer could be realized, for instance, by creating a hierarchical tree of auctioneer nodes in place of a single node that must handle all messages. In this case the leaf nodes receive messages from the trader agents, combine them, and

send them to the next level, and so forth up to the root, with the reverse procedure for return messages. The optimal branching factor for such a tree is e. Each bidding round in the auction involves a message from all the trader agents to the auctioneer, and one in return from the auctioneer to each trader agent. Thus in Table 1 columns 5, 6, 7 and 9. We estimate the message rounds per trading round with such a parallel structure as $2e\ln(N)$ and multiply this by the number of bidding rounds measured in our simulations of the simple centralized system . Figure 10 compares the resulting message round to $\alpha = 2.12$ costs to the peer-to-peer auction. We see a great improvement over the simple single central auctioneer, the distributed auctioneer produces only a logarithmic increase in costs, however we find the peer-to-peer auction still outperforms the centralized version for auctions with more than around 15,000 agents. We further find in Table 1 column 9 that the message rounds per deal after equilibrium also increases logarithmically with a distributed auctioneer. However it this measure still remains higher than in the peer-to-peer system.

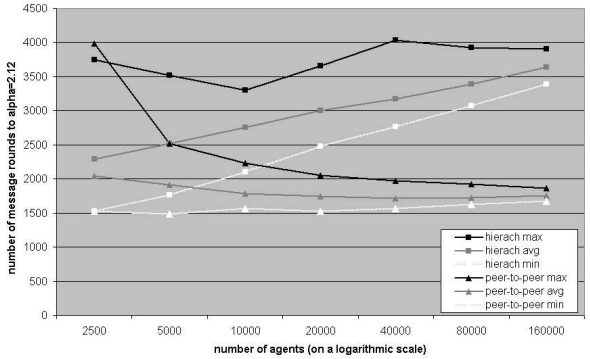

Figure 10: Message rounds to alpha=2.12

We did not measure memory and processing costs in our experiments but we can argue that for any individual entity the comparisons between the peer-to-peer and centralized auctions are similar to those of the message costs. In the centralized auction the auctioneer incurs both high memory and high processing costs, while in the peer-to-peer auction these are distributed among all the agents. The central auctioneer stores bids and offers from all traders each round, costing $O(N)$ memory. Further to calculate deals it needs to sort the lists of bids and offers, costing $O(N \log N)$ time, to find the best ones. As for messages creating a hierarchical auctioneer reduces this sorting cost to $O(\log N)$. Meanwhile, in the peer-to-peer auction the limitations on the basic cluster and clusters sizes keep memory and processing requirements per turn at the cluster centers, the agents that do most of the work, constant.

4. CONCLUSION

In this paper we made use of the fact that in a large market agents have many potential trading partners to create a peer-to-peer auction. We showed that in spite of the fact that trader agents each know only a limited amount of local information, such an auction is able to exhibit market price convergence. Through experiments with a particular agent bidding algorithm and an example supply and demand curve we showed that this peer-to-peer auction has a constant cost in the number of bidding rounds it takes to find equilibrium from a random starting point. While this cost was about two times higher than for a comparison centralized auction where an auctioneer was used to distribute global information, we showed that the cost of reaching equilibrium in terms of message rounds was far better in the peer-to-peer case. For the peer-to-peer auction this message rounds cost remains constant as the number of agents increases while for the centralized case it increases linearly. Even when compared with a distributed hierarchical auctioneer, the peer-to-peer auction showed better performance for systems with 5000 or more agents. The exact numbers presented in this paper depend a great deal upon the bidding strategy of the agents that we choose for our simulations. However, the difference in structure between the peer-to-peer auction and the auction with a centralized auctioneer creates a fundamental difference in the growth rates of the costs of communications which should not change as agents' individual strategies change.

5. REFERENCES

[1] Cliff, D., Bruten, J.: Zero is not enough: On the Lower Limit of Agent Intelligence for Continuous Double Auction Markets. Technical Reoprt HPL-97-141, Hewlett-Packard Laboratories. (1997)

[2] Gode, D., Sunder, S.: Allocative Efficiency of Markets with Zero-Intelligence Traders: Market as a Partial Substitute for Individual Rationality. The Journal of Political Economy, 101:1 (1993) 119-137, 67–79

[3] LeBaron, B.: Agent Based Computational Finance: Suggested Readings and Early Research. Journal of Economic Dynamics and Control 24:5-7 (2000), 679–702.

[4] Ogston, E.,Vassiliadis, S.: Local Distributed Agent Matchmaking. Proceedings of the 9th International Conference on Cooperative Information Systems . (2001) 67–79

[5] Ogston, E.,Vassiliadis, S.: Matchmaking Among Minimal Agents Without a Facilitator. Proceedings of the 5th International Conference on Autonomous Agents.(2001) 608–615

[6] Preist, C., Van Tol, M.: Adaptive Agents in a Persistent Shout Double Auction. Proceedings of the 1st International Conference on the Internet, Computing and Economics. ACM Press (1998) 11–17

[7] Rasmusson, L., Janson, S.: Agents, Self-Interest and Electronic Markets. The Knowledge Engineering Review. 14:2 (1999)143–150

[8] Smith, V.: An Experimental Study of Competitive Market Behavior. The Journal of Political Economy, 70:2 (1962) 111–137

[9] Vulkan, N., Jennings, N.: Efficient Mechanisms for the Supply of Services in Multi-Agent Environments. International Journal of Decision Support Systems, 28:1-2(2000) 5–19

[10] Wellman, M., Walsh, E., Wurman, P., MacKie-Mason, J.: Auction Protocols for Decentralized Scheduling. Games and Economic Behavior 35:1-2 (2001) 271–303