Learning Inventory Management Strategies for Commodity Supply Chains with Customer Satisfaction

Jeroen van Luin[1]
[1]Eindhoven University of Technology
Dep. of Information Systems
Eindhoven, The Netherlands
j.v.luin@tm.tue.nl

Han La Poutré[1,2]
[2]Centre for Mathematics and Computer Science
Amsterdam, The Netherlands
Han.La.Poutre@cwi.nl

J.Will.M. Bertrand[3]
[3]Eindhoven University of Technology
Dep. of Operations, Planning, Accounting and Control
Eindhoven, The Netherlands
J.W.M.Bertrand@tm.tue.nl

ABSTRACT

In this paper, we look at a supply chain of commodity goods where customer demand is uncertain and partly based on reputation, and where raw material replenishment is uncertain in both the amount that is available, as well as the price to pay. Successful participation in such supply chains requires a good inventory management strategy. Actors must find a balance between inventory costs and client satisfaction: structurally high inventory costs reduces the profit, but customers that are faced with a depleted supplier will lose confidence and next time purchase from a competitor. This paper presents a model and a simulation environment to learn successful strategies for participation in this type of supply chains. We combine evolutionary algorithms with logistic theories, and use them in a case in a petrochemical setting. We show that software agents are capable of learning basic and more complex strategies, and that complex learned strategies perform better than basic learned strategies.

1. INTRODUCTION

Supply chain management is an important topic in the area of logistics. The field is still increasing in its relevance and applications, both for industry and the society as a whole. In this field, various mathematical and algorithmic techniques are used[27, 19], typically being part of operations research. The use of agents for studying, optimizing, and implementing supply chains is a rather new and young development. One of the recent developments is e.g. the trading agent competition for supply chain management.

An important aspect of supply chain management is inventory management for the parties in the supply chain. Actors in supply networks are faced with many uncertainties when deciding on a strategy for purchasing their raw materials. They have to find a balance between keeping inventory, with its accompanying inventory costs, and fulfilling the demand of their customers. When structurally keeping an inventory that is larger than required, the costs will reduce the profit. However, when customers are faced with a depleted supplier too often, they can lose confidence and switch to a competitor, thereby decreasing the total sales of the actor, and thus the profit. The aspect of customer satisfaction with respect to inventory management is an important topic in logistics. Several heuristics and methods exist, including look-up tables, but with limited detailed modeling [27]. Therefore, it is important to be able to derive effective inventory management strategies by means of learning agents and agent-based simulations (e.g. like in the field of agent-based computational economics[21, 11, 4].)

In this paper, we consider agents in combination with finding good inventory management strategies for a factory that has a reputation-based share of a stochastic customer base (see section 2). To replenish his inventory, the factory can participate in multi-unit uniform-price auctions[14].

In this paper, we

1. give a model for the supply-market, the demand market with a reputation-based stochastic customer base and the agents that participate in these markets. We base this on agent theory[23], logistics literature[19], client models, and specific types of application cases[13]. This combination enables us to have very realistic models, while keeping a thorough foundation in logistic science.

2. describe the simulation environment we designed, based on this model. The simulation environment allows us to study the market behaviour of software agents in commodity supply chains, together with reputation-based stochastic customer bases.

3. show the results of our experiments with several hard-coded strategies based on logistic theories, and with learning new strategies using adaptive agents. We studied the performance of the strategies in both static and stochastic markets.

4. show that software agents are capable of learning strategies that perform well against theory-based heuristics,

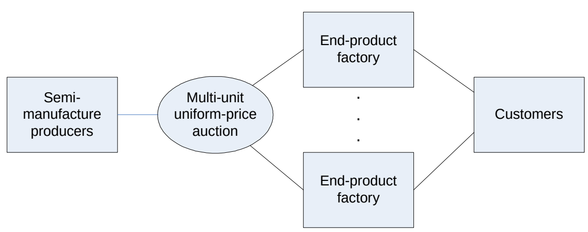
ICEC06, August 14-16, 2006, Fredericton, Canada.

with respect to profit, the competition for market-share, and client satisfaction. We also show that in several settings, complex learned strategies perform better than basic learned strategies.

This paper is structured as follows: in section 2, we give a detailed description of the two markets in which the factory and its competitors participate. Section 3 gives an overview of how our research relates to other work in these fields. The model we designed is described in section 4, the learning algorithm is explained in section 5 and the setup of the simulations and the results are given in section 6. We then summarize our conclusions, and explain our future plans in section 7.

2. DOMAIN

We consider a 4-stage supply chain[19] of commodity goods[1]: raw-material supplier, semi-manufacture producer, end-product factory, and consumer. We focus on the end-product factory stage. As a running example, we look at a case in a petrochemical setting[13]. In this example, our focus stage is the plastics factory, that buys the semi-manufacture (granulates), and sells one or more types of plastic products to its customers.



Each month, these customers place their orders with a factory. The factory has until the start of the next month to produce the products. Customers that get the products they ordered are satisfied with the factory, customers that didn't receive their products are unsatisfied. Unsatisfied orders are not backordered, but lost. The percentage of satisfied customers influences that factory's reputation and, after comparison to the reputation of the competitors, influences the number of customers willing to purchase from the factory in the next month.

The factories can purchase the raw material they need for production through year-contracts and/or monthly auctions. With year-contracts, the factories sign a contract with the semi-manufacture producer for the delivery of semi-manufacture for each of the next 12 months, for a fixed price. The disadvantage of this contract is that once a year, the factories need to make an estimation of their monthly needs for raw materials for each of the next 12 months, and are bound to this contract, even when the estimation turns out to be completely off. The discovery of trends that lead to continuous changes in demand cannot be used until the start of the next year. The advantage of ordering raw materials through year-contracts is the certainty of delivery at the start of each month, for a fixed price. Participating in the monthly auctions has the advantage of knowing how much raw material is needed, but the drawback of uncertainty about the price to pay and amount of the granulates that is acquired. The factories need to find a good purchase strategy in the right combination of both types.

3. LITERATURE REVIEW

The literature relating to multi-agent simulations, multi-unit auctions, repeated games and supply chain management is extensive. In this section, we intent to compare our work to prior research and give examples of current related work. Our research is closely related in subject and methods to the research performed in the Trading Agent Competition (TAC[26]) and the Supply Chain Management game for the Trading Agent Competition (TAC-SCM[1]). In both games, researchers are invited to compete against other agents in trade games. In the TAC, an agent represents a number of travelling clients, each with their own preference on a bundle of services. These services can be obtained in a variety of auctions. The objective of the agent is to purchase valid bundles[2] that are as close a match to the clients' preferences as possible. The TAC-SCM is a competition for trading agents specializing in supply chain management. An agent plays the role of a PC manufacturer, competing against 5 other agents for both customer orders and component procurement, using requests-for-quote[5] in both markets. Production-capacity is limited, and both the customer demand and the component availability can vary. The objective in this game is to make a high profit using a good purchase strategy, production strategy and sell strategy.

Both competition types show several similarities and several differences with our research. Like in our research, the TAC-SCM agents try to find optimal strategies for inventory management in a repeated game. However, both the supply side market and demand side market are quite different. Where the TAC-SCM agents use requests for quote, our agents have to procure their raw-material in single-seller multi-unit uniform-k^{th}-price auctions (see section 4). When the TAC-SCM agents accept a request for quote, they close a contract with the supplier for both amount and price and know exactly how much material they will receive, and what they have to pay. Our agents face an uncertain supply-market. When they make an offer in an auction, they don't know how much, if any, raw material they will receive, and they only know the maximum of the price they have to pay. The TAC-SCM uses customer satisfaction between the raw-material supplier and the agent, where agents that have a high reputation of accepting quotes get a preference in receiving new quotes. Our model uses customer satisfaction as part of the client-model of the agent: a high reputation in delivering the goods to the customer results in a larger share in the demand of the customer base.

The TAC agents share with our agents their need to find an optimal strategy to bid in auctions. The TAC agents need to learn to bid in many parallel auctions, to satisfy given customer needs. Our agents must be able to learn realistic inventory management strategies for an unknown customer demand in repeated auctions. The agents can try to build an inventory at a low price, and use the inventory

[1]Goods that are traded on the basis of price, not on differences in quality or features

[2]Regardless of client preference, the bundles must follow certain rules before they are considered valid.

when market-prices are high, or to compensate for the uncertainty in the amount of customer orders. Markets with an uncertain supply amount, an uncertain supply price and an uncertain demand that is influenced by reputation, make a more difficult, but also more realistic setting, based on logistic literature and market literature for commodity goods.

Many of the participants in the TAC and TAC-SCM have an active interest in trading agents in general. The AI laboratory of Wellman, the founder of the TAC, works on automated markets[15, 18, 25] and strategic reasoning[24]. The Intelligence, Agents and Multimedia Group of the University of Southampton, supervised by Jennings, is active in the fields of adaptive behavior[10, 12] and inter-agent negotiation[3, 9].

In the field of supply chain management, and more specifically in inventory control, we base our work on the book of Silver, Pyke and Peterson[19], and the work of Zipkin[20]. Cachon[6, 7] takes a more game-theoretic approach to inventory management.

4. MODEL

In our model, we have a set $B = \{b_1 \dots b_n\}$ of n competitive buyer agents representing the factories in the supply chain. The environment further contains an auctioneer agent and a client base. In the model, time is divided into rounds, each representing one month.

4.1 Auction

In each round the agents are informed of the amount of customers that are willing to buy a product from them, the amount of raw material units A that are available in the auction[3] and a reserve price per unit p_{res}. Each agent b_x then informs the auctioneer of his bid (a_x, p_x) containing an amount of units of raw material $0 \leq a_x \leq A$ and a price per unit $p_x \geq p_{res}$. The auctioneer determines the set of agents $W \subseteq B$ that win and for each $w \in W$ the amount of raw material units won_w he won. The bidder with the highest price per unit receives the amount he asked for. Then the second-highest bidder receives his share, and so on, until either all bidders received they amount the asked for, or the supply of raw material is completely allocated, whichever comes first. When the last share to be allocated contains less material than the agent who is entitled to it asked for, the agent has to accept the smaller share. We use a uniform-k^{th}-price auction[14], which means that the price q to pay is $max\{p_l \mid l \in B \setminus W\}$, the highest price offered by agents that did not win anything. In the case that all agents won something, $W = B$, the price to pay $q = p_{res}$.

4.2 Inventory management

The amount of raw material needed by each agent b_x in a round t depends on the size of the current inventory $inv_x^t \geq 0$, the chosen inventory management technique, the amount of raw materials ordered in the year-contract, and the customer demand for this round. In our model, we combine the year-contract amount and the customer demand into one value d_x^t, representing how much the amount of raw-material needed to satisfy the demand for month t deviates from the amount ordered in the year-contract. A value of $d_x^t < 0$ means that the demand needs less raw material than the amount in the year-contract, and that part of the raw material material from the year-contract must be added to the inventory. With $d_x^t > 0$, the demand requires more raw material than the amount in the year-contract, and the agent has to use his inventory (if any) or buy raw material in the auction in order to satisfy the demand. Even if the current inventory is sufficient to satisfy the customer demand, the agent can choose to buy raw material to build inventory, knowing that it can afford to bid a low price. As mentioned, any unsatisfied demand is lost, and also results in a loss in client satisfaction.

To keep track of the performance of the agents in the market, we do some bookkeeping[19]. To calculate the profit $profit_x^t$ for agent x in a round t, we need the accumulative revenue $revenue_x^t$, the holding costs per round per unit h, the accumulative holding costs hc_x^t, the accumulative purchase costs pc_x^t, the amount of products $sold_x^t$ sold in this round, and the fixed retail price per product r_x^t. We use the following formulas:

- $sold_x^t = min(d_x^t, inv_x^t + won_x^t)$
- $pc_x^t = pc_x^{t-1} + (won_x^t * q_x^t)$
- $hc_x^t = hc_x^{t-1} + h * (inv_x^t + won_x^t - sold_x^t)$
- $revenue_x^t = revenue_x^{t-1} + (sold_x^t * r_x^t)$
- $profit_x^t = revenue_x^t - pc_x^t - hc_x^t$

At the end of a round, we can calculate the inventory left for the next round:

- $inv_x^{t+1} = inv_x^t + won_x^t - sold_x^t$

where we assume that the inventory can hold an unlimited number of items.

4.2.1 Inventory Management Techniques

In conventional base-stock inventory management techniques [8, 19], the agent defines a reorder point[22] rp. Whenever an agent b_x finds that the inventory drops below this reorder point, he orders enough items to increase the inventory to the reorder point:

$$a_x^t = max(0, d_x^t + (rp_x - inv_x^t))$$

In function-based inventory management techniques, the amount and the price offered by an agent are functions of the current inventory size and the current demand. We use N-point piece-wise linear functions for both the amount and the price. We represent these by the set of N (inventory-size, amount, price)-tuples:

$$\{(\sigma_0, \alpha_0, \pi_0), (\sigma_1, \alpha_1, \pi_1), \ \dots \ , (\sigma_N, \alpha_N, \pi_N)\}, \quad \sigma_0 \leq \sigma_1 \leq \dots \leq \sigma_N$$

[3]For easy calculation, all products the agents can produce require one unit of raw material to produce.

which defines the amount a_x^t offered by agent x in round t as:

$$a_x = \begin{cases} \frac{inv_x}{\sigma_0}\alpha_0, & \text{if } inv_x < \sigma_0 \\ \alpha_k, & \text{if } inv_x = \sigma_k; \forall k \in [0, N] \\ \alpha_k + \frac{(inv_x - \sigma_k)}{(\sigma_{k+1} - \sigma_k)}(\alpha_{k+1} - \alpha_k) & \text{if } \sigma_k \leq inv_x \leq \sigma_{k+1}; \\ & \forall k \in [0, N-1] \\ \alpha_N, & \text{if } inv_x > \sigma_N \end{cases}$$

and when we substitute π_i for α_i, we get the price p_x^t offered by agent x in round t.

4.2.2 Client Satisfaction

As mentioned in section 2, the demand for an agent x is influenced by his reputation: the client satisfaction ratio of the agent, compared to that of his competitors. We can define the client satisfaction $cs_x^t \in [0, 1]$:

- $cs_x^t = \begin{cases} 1, & \text{if } t = 0 \\ \frac{\sum_{i=0}^{t-1} sold_x^i}{\sum_{j=0}^{t-1} d_x^j}, & \text{if } t > 0 \end{cases}$

The demand also consists of a part that is independent of reputation: the constant loyal customers lc_x of agent x, for whom the reputation of an agent is not the dominant factor in their choice where to buy. Our client statisfaction model is an abstract model, that captures the basic aspects of customer satisfaction. With constant loyal customers, our model is completed as a consistent and stable model.

We take D^{max} as the maximum total demand for any round, and $D^t \in [0, D^{max}]$ the actual total demand for a round t. The demand d_x^t for agent x in a round t is then defined as:

- $d_x^t = lc_x + \frac{cs_x^t}{\sum_{i=1}^{n} cs_i^t} \quad D^t - \sum_{j=1}^{n} lc_j$

5. LEARNING ALGORITHM

We will use a genetic algorithm [16] with a linear ranking and elitist selection[2], to let adaptive agents learn purchase strategies. In this genetic algorithm, a population of pop_{max} strategies is randomly generated and each strategy is used by an agent in a simulation of the inventory model. In the simulation, a learning agent competes with $(n-1)$ competitors that use fixed values, a heuristic, or another adaptive algorithm to participate in the auctions. When all strategies have been used in a simulation, the population is ranked by profit, the best strategy is copied to the next generation, and the other $(pop_{max}-1)$ places in the population are filled using a linear ranked selection process of two parent strategies, and the use of recombination and mutation to arrive at an offspring strategy. This process is repeated for gen_{max} generations.

The chromosome used for the learning agent depends on the chosen inventory management technique. In case of base-stock inventory management, the chromosome for an agent b_x consists of:

- Reorder-point $rp_x \in \mathbb{N}_0$
- Price $p_x \in \mathbb{N}$
- Reorder-point-mutation size $rm_x \in \mathbb{Z}$
- Price-mutation size $pm_x \in \mathbb{Z}$
- Mutation probability $prob_x \in [0, 1]$

In the case of the function-based inventory management, we use the following chromosome:

- Inventory point $i_{xj} \in \mathbb{N}_0$, for $j \in 1, 2, ..., N$
- Amount point $a_{xj} \in \mathbb{N}_0$, for $j \in 0, 1, ..., N$
- Price point $p_{xj} \in \mathbb{N}$, for $j \in 0, 1, ..., N$
- Inventory mutation size $im_x \in \mathbb{Z}$
- Amount mutation size $am_x \in \mathbb{Z}$
- Price mutation size $pm_x \in \mathbb{Z}$
- Mutation probability $prob_x \in \mathbb{R} \cap [0, 1]$

With both techniques, the genetic algorithm creates the offspring chromosome by choosing randomly, for each part of the chromosome, the corresponding value of one the parents. When the new chromosome is created, the mutation probability and mutation sizes are used to mutate the values of the reorder-point and price (in case of the base-stock technique) or inventory-point, amount, and price (in case of the function-based technique). Finally, the mutation sizes and mutation probability themselves are mutated with small, fixed values, randomly up or down. For the base-stock agents, the process of evolution is depicted in figure 1.

6. EXPERIMENTAL SETUP AND RESULTS

In our experiments, we first use a number of different heuristic competitor strategies to test our learning agents against. In the following section, we choose b_n as our learning agent, and $b_1 \dots b_{n-1}$ as the heuristic competition. We continue the experiments with simulations in which multiple adaptive agents compete against eachother, in coevolving simulations.

6.1 Agents with a base-stock inventory management

6.1.1 Model

We setup our experiments with the following parameter values for each simulation:[4]

- Number of agents $n = 5$
- Number of rounds $T = 500$

[4] As an abbreviation, we use the following notation:

Uniform$_{\mathbb{Z}}$(x,y)	denoting a function drawing values uniformly from $\mathbb{Z} \cap [x, y]$.
Uniform$_{\mathbb{R}}$(x,y)	denoting a function drawing values uniformly from $\mathbb{R} \cap [x, y]$.

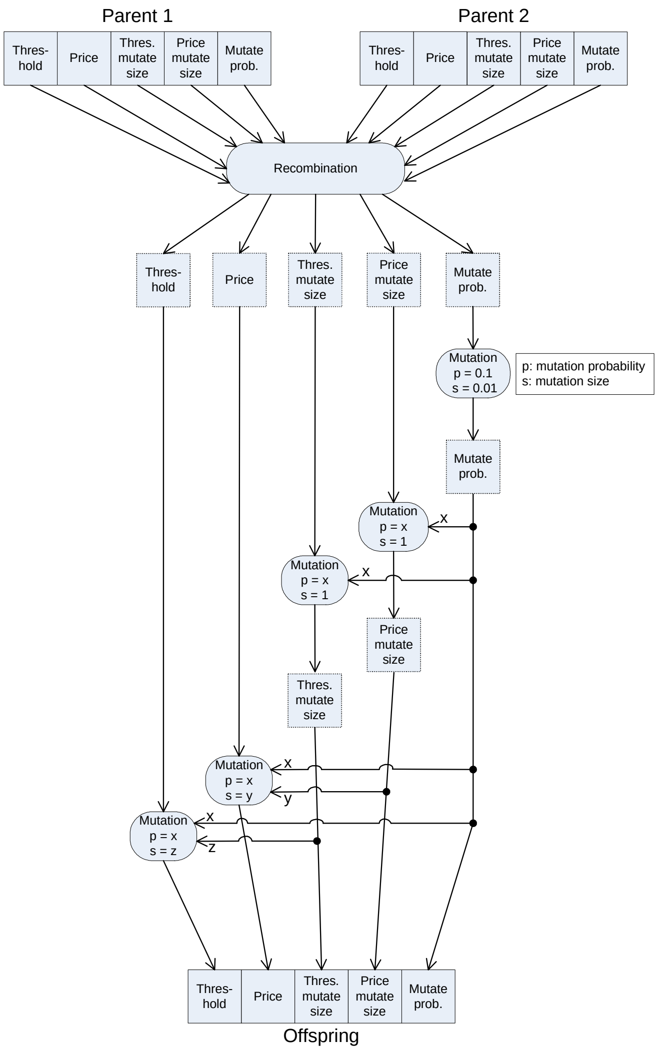


Figure 1: The evolutionary algorithm for an agent with a base-stock inventory management.

- Amount sold in auction $A = 50$
- Reserve price $p_{res} = 1$
- Revenue per item sold $r_x^t = 20, \forall x \in [1, n]; \forall t \in [0, T]$
- Holding costs per item per round $h = 1$
- Maximum global demand $D_{max} = 55$
- Monthly demand $D^t = \text{Uniform}_{\mathbb{Z}}(n, D_{max}), \forall t \in [0, T]$
- Constant loyal customers $lc_x = 0, \forall x \in [1, n]$

In this first simulation, there are no constant loyal customers, and the demand share of the agents depends completely on their customer satisfaction.

According to [27], we can calculate a good practice for the level of the base-stock, if we know the mean absolute deviation of the demand[5], desired customer service level[6] and the lead time. The function given in [27] for calculating base stock is:

$$\text{base-stock} = \text{Cust. serv. factor} \times \text{MAD} \times \sqrt{\text{lead time}}$$

where the customer service factor is a factor depending on the desired service level. For a service level of 50%, the factor is 0, for 80% the factor is 1, for 90% it is 1.6 and for 99.99% the factor is 5 (see [17]). With a monthly demand uniformly distributed between 5 and 55, the MAD = 12.7; divided between 5 agents. When we use each of the service levels for one of the heuristic agents, we get the following base-stock levels:

Agent	Service level	Reorder-point
b_1	50%	$rp_1 = 0$
b_2	80%	$rp_2 = 3$
b_3	90%	$rp_3 = 4$
b_4	99.99%	$rp_4 = 13$

The heuristic agents $b_1 \ldots b_{n-1}$ use the following algorithm to compute the price to bid each round:

$$p_x^0 = \frac{r_x^t + p_{res}}{2}$$

$$p_x^t = \begin{cases} p_x^{t-1} - 1 & \text{if } (i_x^{t-1} > 0) \wedge (won_x^{t-1} > 0) \\ p_x^{t-1} + 1 & \text{if } (i_x^{t-1} > 0) \wedge (won_x^{t-1} = 0) \\ p_x^{t-1} & \text{otherwise} \end{cases}$$

In summary, the agent starts with a price halfway between the reserve price and the highest price that would give him a profit. The agent decreases the price offered if in the previous round, he participated in an auction ($i_x^{t-1} > 0$) and won ($won_x^{t-1} > 0$). When he participated but lost ($won_x^{t-1} = 0$), the price for this round is increased. If he didn't participate in an auction in the previous round, the price stays the same.

6.1.2 Genetic Algorithm

The genetic algorithm for learning agent b_n used these parameters:

- Population size $pop_{max} = 100$
- Number of generations $gen_{max} = 300$
- Initial reorder-point $rp_n^0 = \text{Uniform}_{\mathbb{Z}}(0, 100)$
- Initial price $p_n^0 = \text{Uniform}_{\mathbb{Z}}(1, 20)$
- Init. rp_n-mutation size $rm_n^0 = \text{Uniform}_{\mathbb{Z}}(-25, 25)$
- Init. p_n-mutation size $pm_n^0 = \text{Uniform}_{\mathbb{Z}}(-5, 5)$
- Init. mutation prob. $prob_n^0 = \text{Uniform}_{\mathbb{R}}(0, 1)$

6.1.3 Results

The reorder-point and price strategy learned by the adaptive agents in each of the generations in our simulation is shown in figures 2 to 6.

[5]the mean of the absolute deviation from a set of values to the mean of that set:
$MAD(x) = \frac{1}{N}\sum_{i=1}^{N}|x_i - \overline{x}|$

[6]The desired service level is the percentage of customer demand that on average should be satisfied.

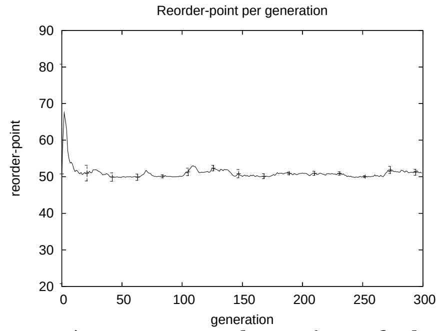


Figure 2: Average reorder-point of the adaptive agent, per generation, in a market without constant loyal customers.

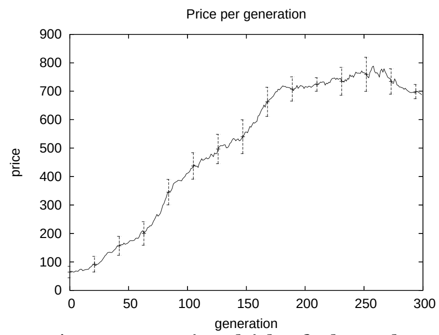


Figure 3: Average price bid of the adaptive agent, per generation, in a market without constant loyal customers.

As can be seen from the pictures, the adaptive agent learned to keep a reorder-point just above 50, the amount of items for sale. The offered price didn't converge, and was high enough to always be a winning bid. Of course, as our model uses a uniform-price auction, the price that is actually paid is much lower, as long as either all agents win at least one unit of raw material, or at least one agent who didn't win anything bid a low price (see section 4.1), due to the auction type. The continuously upward direction of the price graph can be attributed to the fall-off of the lower values, a property inherent to our AI approach. The profit averages just above 200,000; but it is obvious that the agent cannot rely on making this profit: the variation in the profit is very high.

Because of the reputation-dependent demand function, and the absence of constant loyal customers ($lc_x = 0$), the adaptive agent learned to push his competitors out of the market, almost creating a monopoly, as can be seen from the market-share figure (figure 5). The agent couldn't afford to create a complete monopoly: the second-price auction mechanism requires the agent to keep at least one competitor that bids a low price, to avoid having to pay his own, very high, price.

We repeated the simulation, this time with a small constant loyal demand: $lc_x = 1$. All agents are ensured that they get at least one customer each round, giving them a possibility to improve their reputation. The results of this experiment can be seen in figures 7 to 11.

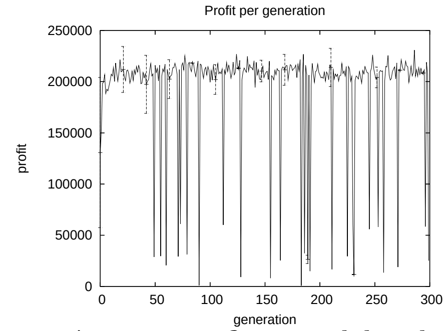


Figure 4: Average profit earned by the adaptive agent, per generation, in a market without constant loyal customers.

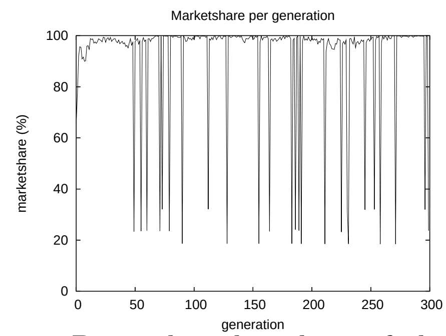


Figure 5: Demand-market share of the adaptive agent, per generation, in a market without constant loyal customers.

Without the possibility to remove his competitors from the market, the agent faces a market that is much more uncertain in who wins, and for what price. This increased uncertainty can be seen in the figures as a much lower reorder-point: the agent behaves as if the demand market was split evenly among the five agents. The market-share figure (figure 10) shows that this is almost the case: the adaptive agent reached a demand-market-share of 22%.

Keeping the reorder-point low apparently ensures that there is always more supply than demand (keeping the price-to-pay at the reserve price). This allows the agent to bid extremely high (as can be seen in figure 8), without running the risk of paying that price. The profit earned by the adaptive agent averages between 38,000 and 40,000; a lot less than when he was able to monopolize the market. There is still a lot of variation in the profit.

6.2 Coevolution learning with base-stock inventory management agents

In our next simulation, we let several adaptive agents learn simultaneously, while competing against each other.

6.2.1 Model

The model for the coevolution learning simulations is similar to that of the simulations with one learning agent, as described in section 6.1.1, but this time all agents are learning.

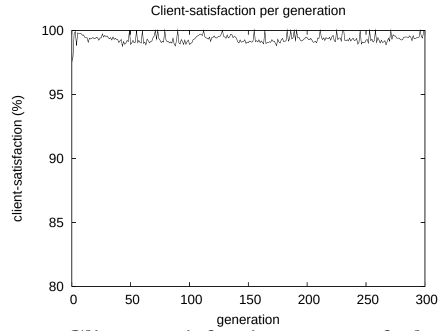


Figure 6: Client-satisfaction rate of the adaptive agent, per generation, in a market without constant loyal customers.

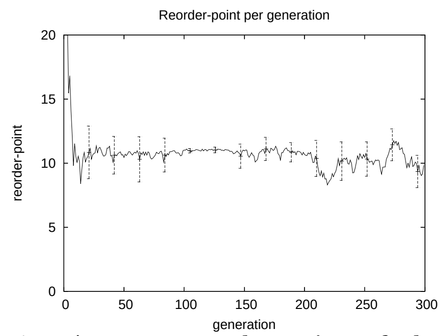


Figure 7: Average reorder-point of the adaptive agent, per generation, in a market with loyal customers.

6.2.2 *Genetic Algorithm*

The genetic algorithm used for coevolution is an extension of the algorithm used in the single-agent learning simulations (see sections 5 and 6.1.2). Instead of one population of agents, we use n populations, one for each of the participants in the auction.

Within one generation, each agent strategy from one population is used 5 times, each time against different strategies from the other populations. By making sure that no strategy is used more than 5 times, we know that after $5 * pop_{max}$ simulations, each strategy in each population in that generation has been used in 5 simulations. Each population then simultaneously, but independently, evolves to the next generation, where the process is repeated.

6.2.3 *Results*

The entire experiment as described above is repeated 5 times, and the results are averaged. Figures 12 to 16 show the results from the simulations.

The figures show that the agents were able to learn a profitable strategy. They reached an even-split situation, dividing the supply in almost equal shares. With a stochastic demand for each agent in each round, it is very probable that not all agents require a complete refill of their inventory. When at least one agent needs less than his reorder-level, all other agents can bid for a amount equal to their reorder-level, and all agents still pay the reserve price. Because of this strategy, the agents can afford to bid any price

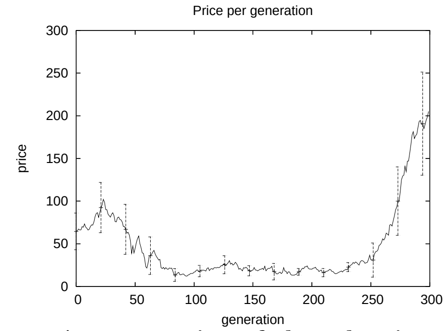


Figure 8: Average price of the adaptive agent, per generation, in a market with loyal customers.

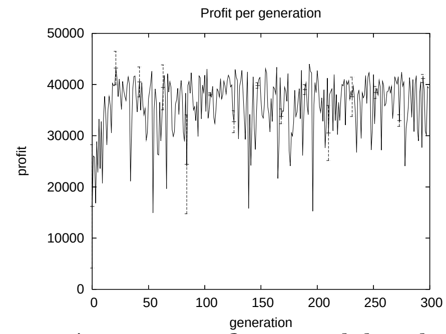


Figure 9: Average profit earned by the adaptive agent, per generation, in a market with loyal customers.

they want, and never pay more than the reserve price, a behaviour that we clearly see in the figure of the price function (figure 13).

6.3 Agents with a function-based inventory management

Similar to our experiments with base-stock agents, we start with one adaptive agent competing against a number of heuristic agents. The function-based technique allows agents to let both the amount and the price in his offer, to be dependent on the current inventory (see section 4.2.1). This allows agents to try to buy cheap raw material, even when their inventory is already sufficient to satisfy their demand.

6.3.1 *Model*

We setup our experiments for the function-based inventory management with the following parameter values:

- Number of agents $n = 5$
- Number of rounds $T = 500$
- Amount sold in auction $A = 50$
- Reserve price $p_{res} = 1$
- Revenue per item sold $r^t = 20$
- Holding costs per item per round $h = 1$
- Maximum global demand $D_{max} = 55$
- Monthly demand $D^t = \text{Uniform}_{\mathbb{Z}}(n, D_{max})$

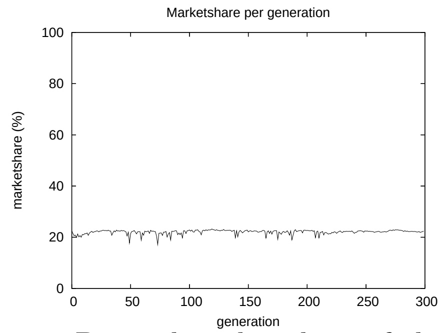


Figure 10: Demand-market share of the adaptive agent, per generation, in a market with loyal customers.

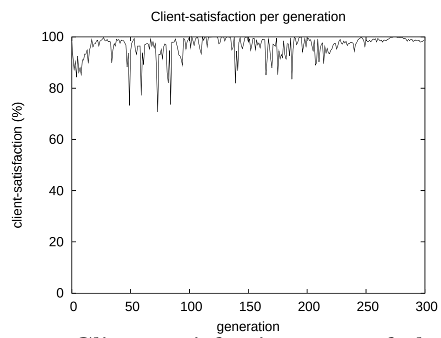


Figure 11: Client-satisfaction rate of the adaptive agent, per generation, in a market with loyal customers.

- Loyal customers $lc_x = 1, \forall x \in [1, n]$
- Number of function steps $N = 5$

The heuristic agents are identical to the heuristic agents used in section 6.1.

6.3.2 Genetic algorithm

The genetic algorithm for the function-based agent simulations use the following parameter values:

- Population size $pop_{max} = 100$
- Number of generations $gen_{max} = 300$
- Initial inventory point value $i^0_{xj} = \text{Uniform}_{\mathbb{Z}}(0, 50)$, $\forall j \in \{1, \ldots, N\}$
- Initial amount point value $p^0_{xj} = \text{Uniform}_{\mathbb{Z}}(0, 50)$, $\forall j \in \{1, \ldots, N\}$
- Initial price point value $a^0_{xj} = \text{Uniform}_{\mathbb{Z}}(0, 50)$, $\forall j \in \{1, \ldots, N\}$
- Initial inventory mutation size $im^0_x = \text{Uniform}_{\mathbb{Z}}(-25, 25)$
- Initial amount mutation size $am^0_x = \text{Uniform}_{\mathbb{Z}}(-25, 25)$
- Initial price mutation size $pm^0_x = \text{Uniform}_{\mathbb{Z}}(-25, 25)$
- Initial mutation probability $prob^0_x = \text{Uniform}_{\mathbb{R}}(0, 1)$

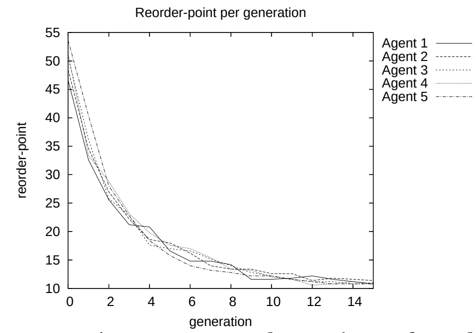


Figure 12: Average reorder-point of co-learning agents, per generation.

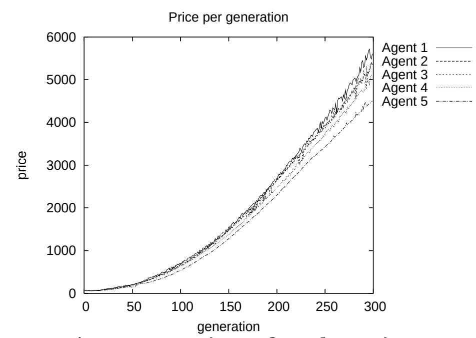


Figure 13: Average price of co-learning agents, per generation.

6.3.3 Results

The results of the simulations with an agent with a function-based inventory management, without constant loyal customers ($lc_x = 0$), and with constant loyal customers ($lc_x = 1$) are depicted in figures 17 to 22.

In these figures, the functions for amount and price are those of the best performing agents in the simulations. To be able to compare the performance with those from the base-stock agents, the profits for both the markets with constant loyal customers, and without constant loyal customers are averaged over the entire population, similar to the results from the base-stock agent experiments.

We see from figures 17 and 20 that the function-based agent learns to behave like a base-stock when the inventory is empty. When the inventory is non-empty, the function-based agent buys more than the base-stock agent would. The base-stock agent buys enough to fill the inventory up to the reorder point (on average 51 for the market without constant loyal customers, on average 11 for the market with constant loyal customers), the function-based agent's inventory size plus the amount he orders exceeds these values, for all values of the inventory-size. This strategy turns out to be more successful than that of the base-stock agent. In the market without constant loyal customers, the averaged profit of the function-based agent is around 240000, as opposed to around 220000 for base-stock agents. In the market with constant loyal customers, the averaged profit is around 42000, as opposed to around 39000 for the base-stock agents.

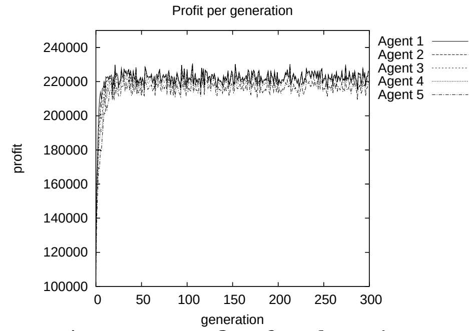


Figure 14: Average profit of co-learning agents, per generation.

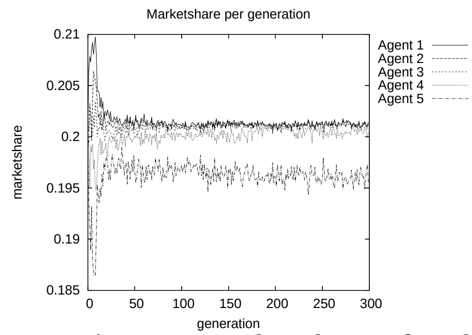


Figure 15: Average market-share of co-learning agents, per generation.

Figure 16: Average client satisfaction of co-learning agents, per generation.

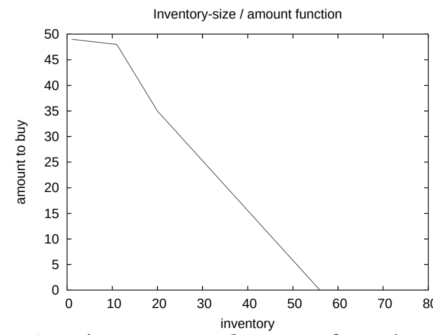


Figure 17: Amount to buy as function of the inventory size for a function-based agent in a market without constant loyal customers.

6.4 Coevolution learning with a function-based inventory management

Our last set of experiments presented in this paper let agents with a function-based inventory management technique compete and learn simultaneously.

6.4.1 Model

The model for the coevolution learning simulations with a function-based inventory management is similar as to that of the simulations with only one learning agent.

6.4.2 Genetic Algorithm

The genetic algorithm for coevolution with function-based agents is an extension to the algorithm described in section 6.3.2. The extensions are similar to the extensions used in coevolution with base-stock agents (section 6.2.2).

6.4.3 Results

In figures 23, 24 and 25 we show the results of coevolutionary learning of 5 agents with function-based inventory management. The best-performing strategy from each population is chosen as that population's winner. Figure 23 shows the functions learned by the 5 winning agents, for calculating how much to buy, depending on the size of the inventory. In figure 24 we show the learned functions for the price to bid, depending on the size of the inventory. The third figure, figure 25 shows the profit earned by the agents, averaged over the populations, to allow us to compare the results to those of the coevolving base-stock agents.

As we can see from figure 25, the coevolution learning agents with function-based inventory management succeeded in learning succesful strategies. The profit of the coevolving base-stock agents averaged around 220000, the profit of three of the five evolving function-based agents averaged around 230000.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have given a model based on agent theory, logistics literature, client models, and an application case, for participating in a supply chain of commodity goods with uncertain, reputation-based customer demand and uncertain raw-material replenishment. We describe the simulation environment we designed, based on this model, and showed the results of our experiments with several hard-coded strategies and with (co)learning agents, using two types of inventory management. We have shown that software agents are capable of learning successful strategies, with respect to profit, the competition for market-share, and client satisfaction. We have also shown that more complex inventory management techniques perform better than standard inventory management techniques in several settings.

In our future research, we will improve the coevolution algorithm for function-based inventory management agents, to prevent the premature convergence that lowers its performance. We will also add the possibility for the function-based agents to submit a list of bids, instead of just one bid. This will allow the agents to differentiate between amounts of raw-material that they really need (by offering a high price), and raw-material that they can get cheap, and use when market prices are high.

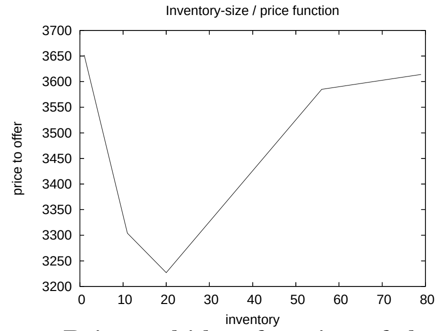


Figure 18: Price to bid as function of the inventory size for a function-based agent in a market without constant loyal customers.

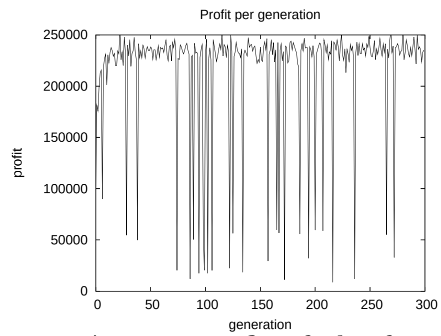


Figure 19: Average profit of the function-based agents in a market without constant loyal customers.

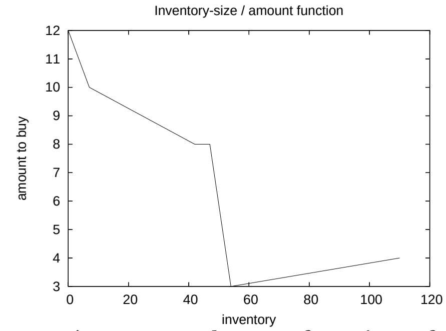


Figure 20: Amount to buy as function of the inventory size for a function-based agent in a market with constant loyal customers.

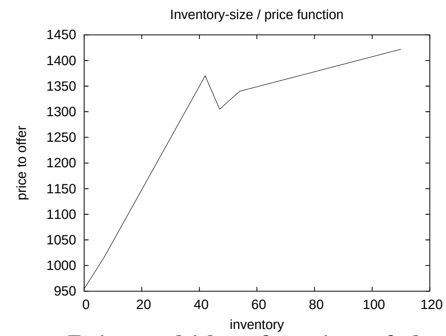


Figure 21: Price to bid as function of the inventory size for a function-based agent in a market with constant loyal customers.

A third extension will be to allow for multiple types of products that can be ordered by the customers, and multiple types of raw-material needed to produce them. We will include the possibility that one of the products is 'in fashion', and the agents must learn whether it is more profitable to specialize in one item, or to stay generalized and target a broader set of customers.

8. REFERENCES

[1] R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne, and S. Janson. The supply chain management game for the trading agent competition 2004. Technical report, ISRI, Carnegie Mellon University and Swedish Institute of Computer Science, 2004.

[2] T. Bäck and F. Hoffmeister. Extended selection mechanisms in genetic algorithms. In R. Belew and L.B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 92–99, 1991.

[3] C. Bartolini, C. Priest, and N.R. Jennings. A software framework for automated negotiation. *Software Engineering for Multi-Agent Systems III: Research Issues and Practical Applications*, pages 213–235, 2005.

[4] S.M. Bohté, E.H. Gerding, and J.A. La Poutré. Market-based recommendation: Agents that compete for consumer attention. *ACM Transactions on Internet Technology (ACM TOIT)*, 4:420 – 448, 2004.

[5] S. Buffett and N. Scott. An algorithm for procurement in supply-chain management. *Workshop on Trading Agent Design and Analysis (TADA04)*, 2004.

[6] G. Cachon and M. Lariviere. Supply chain coordination with revenue sharing contracts. *Management Science*, 51(1):33–44, 2005.

[7] G. Cachon and P.H. Zipkin. Competitive and cooperative inventory policies in a two-stage supply chain. *Management Science*, 45(7):936–953, 1999.

[8] M. Ettl, G.E. Feigin, G.Y. Lin, and D.D. Yoa. A supply network model with base-stock control and service requirements. *Operations Research*, 48(2):216–232, 2000.

[9] S. Fatima, M. Wooldridge, and N.R. Jennings. Bargaining with incomplete information. *Annals of Mathematics and Artificial Intelligence*, 44(3):207–232, 2005.

[10] S. Fatima, M. Wooldridge, and N.R. Jennings. A comperative study of game theoretic and evolutionary models for software agents. *Artificial Intelligence Review*, 23(2):187–205, 2005.

[11] E.H. Gerding and J.A. La Poutré. Bargaining with posterior opportunities: An evolutionary social simulation. In M. Gallegati, A.P. Kirman, and M. Marsili, editors, *The Complex Dynamics of Economic Interactions*, volume 531 of *Springer LNEMS*. Springer Verlag, 2003.

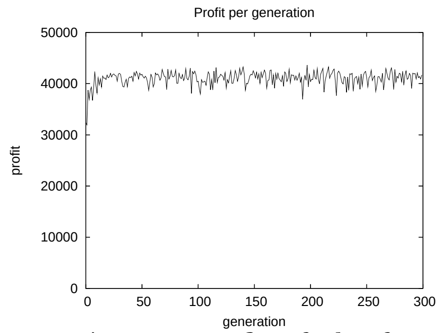


Figure 22: Average profit of the function-based agents in a market with constant loyal customers.

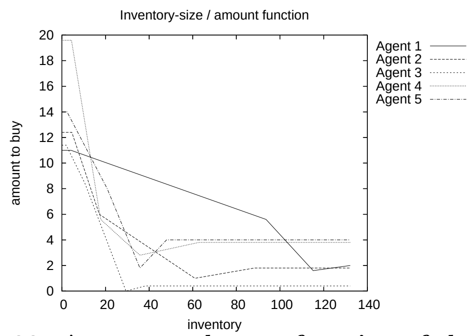


Figure 23: Amount to buy as function of the inventory size, for 5 co-learning agents with a function-based inventory-management strategy.

Figure 24: Price to bid as function of the inventory size, for 5 co-learning agents with a function-based inventory-management strategy.

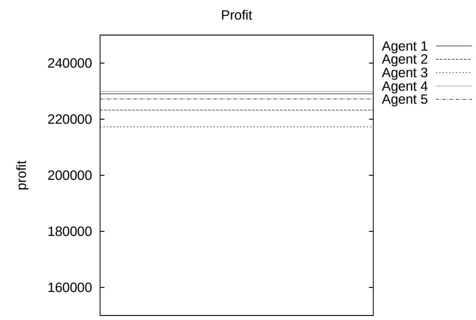


Figure 25: Average profit for 5 co-learning agents with a function-based inventory-management strategy.

[12] M. He, N.R. Jennings, and A. Prugel-Bennett. An adaptive bidding agent for multiple english auctions: a neuro-fuzzy approach. In *Proceedings of IEEE Conference on Fuzzy Systems*, pages 1519–1524. Budapest, Hungary, 2004.

[13] C.A.M. Hurkens. De seriegroottebeslissing (production-size decisions). Masters thesis, Technische Universiteit Eindhoven, 2002. (in Dutch).

[14] J.H. Kagel and D. Levin. Behavior in multi-unit demand auctions: experiments with uniform price and dynamic vickrey auctions. *Econometrica*, 69(2):413–454, March 2001.

[15] J.K. MacKie-Mason and M.P. Wellman. Automated markets and trading agents. Technical report, University of Michigan, April 2005.

[16] M. Mitchell. *An introduction to genetic algorithms.* MIT Press, 1996.

[17] G.W. Plossl. *Production and Inventory Control, Principcles and Techniques.* Prentice Hall, 1985.

[18] D.M. Reeves, M.P. Wellman, J.K. MacKie-Mason, and A. Osepayshvili. Exploring bidding strategies for market-based scheduling. *Decision Support Systems*, 39:37–85, 2005.

[19] E.A. Silver, D.F. Pyke, and R. Peterson. *Inventory Management and Production Planning and Scheduling.* John Wiley & Sons, 3rd edition, 1998.

[20] J. Song and P. Zipkin. Supply-chain operations: assemble-to-order systems. In A. de Kok and S. Graves, editors, *Handbooks in OR/MS 30: Supply Chain Management*, 2003.

[21] L. Tesfatsion. Introduction to the special issue on agent-based computational economics. *Journal of Economic Dynamics and Control*, 25(3-4):281–293, 2001.

[22] John Toomey. *Inventory Management, Principles, Concepts and Techniques.* Kluwer Academic Publishers, 2000.

[23] G. Weiss (editor). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence.* The MIT Press, 1999.

[24] M.P. Wellman, J. Estelle, S. Singh, Y. Vorobeychik, C. Kiekintveld, and V. Soni. Strategic interactions in a supply chain game. *Computational Intelligence*, 21(1):1–26, 2005.

[25] M.P. Wellman, D.M. Reeves, K.M. Lochner, and Y. Vorobeychik. Price prediction in a trading agent competition. *Journal of Artificial Intelligence Research*, 2004.

[26] M.P. Wellman and P.R. Wurman. A trading agent competition for the research community. Technical report, University of Michigan, 1999.

[27] Tony Wild. *Best Practice in Inventory Management.* Elsevier Science Ltd, 2nd edition, 2002.