Fuzzy adaptive agent for supply chain management

Yain-Whar Si [a,*] and Sio-Fan Lou [b]
[a,b] *Faculty of Science and Technology, University of Macau, Av. Padre Tomas Pereira, Taipa, Macau*

Abstract. Recent technological advances in electronic commerce have fuelled the need for designing effective strategies for supply chain management. These strategies are essential in guiding various activities within a supply chain such as component acquisition, inventory management, customer orders bidding, scheduling for production, and delivery. Among these activities, planning of raw material acquisition for inventory and bidding for customer orders can be extremely complex since such activities are required to deal with external entities such as suppliers and customers. Specifically, in the context of the manufacturing industry, acquiring raw materials for maintaining a flexible yet adequate inventory level is a complex issue due to the potential fluctuation in suppliers' production capacity and market demand. Designing an effective strategy for bidding customer orders is also an intricate problem due to the intense competition in fast changing market environments. In this paper, we describe the strategies of a supply chain management agent which adaptively adjusts its target inventory level and customer order bidding price based on fuzzy logic reasoning. The agent has competed in the 2006 Trading Agent Competition for Supply Chain Management and has achieved good results.

Keywords: Intelligent agents, supply chain management, fuzzy logic, target inventory level, bidding price setting, Trading Agent Competition

1. Introduction

A supply chain is considered a network of entities and their related activities that work together to produce value for the customer [23]. Supply chain management involves acquisition of raw materials, assembling of finished products, and delivering these products to customers [5]. These tasks are closely intertwined, in that shortcomings in one can adversely affect the others. For instance, intense competition among manufacturers who are seeking to acquire raw materials may drive suppliers to reduce their usual capacity allocated to their loyal customers. It is crucial that such a situation is constantly monitored by manufacturers and reflected in their business strategies. In such circumstances, manufacturers may need to immediately increase the components bidding price for securing stable supplies. It is also crucial that manufacturers take into account the available inventory level and market demand in setting appropriate selling prices for the assembled products. If the market is inundated with oversupply from competitors, a manufacturer may need to immediately re-adjust its selling price and production level. In today's complex e-business environment, supply chain managers face constant pressure to respond to changes in the environment as quickly as possible. However, immediate response to the market situation cannot be easily carried out by managers and business analysts since such variations in market condition can be subtle (gradual) and hard to recognize. In this paper, we investigate how manufactures and supply chain mangers can efficiently respond (adapt) to the market situation through a simulated agent-based supply chain management environment.

In the past decade, agent technology has been extensively used to model supply chain components. The generic properties of software agents [24] such as autonomy, reactivity, pro-activeness, and social ability are well suited to capture the characteristics of complex supply chain components and their interaction activities. Specifically, a multi-agent approach to

[*] Corresponding author. E-mail: fstasp@umac.mo. Phone: (853) 83974355, Fax: (853) 28838314

modeling supply chain management has been extensively adopted in a number of research prototypes.

One of the major applications of agent-based modeling is to simulate supply chain activities in a virtual environment. In a framework proposed by Swaminathan et al. [14], supply chain models are composed from software components that represent types of supply chain agents (e.g. retailers, manufacturers, transporters), their constituent control elements (e.g., inventory policy), and their interaction protocols (e.g., message types). In a similar framework [26], supply chain elements such as flows, material, and information are modeled as objects and software agents are used to emulate entities such as enterprises and internal departments. The framework proposed in [26] is then used as a decision support system to perform simulation analysis for petrochemical cluster management and crude procurement in a refinery [26]. A multi-agent approach is also used to simulate the effect of different scheduling algorithms under varying orders in supply chains [38].

Multi-agent systems are also extensively used for modeling and analysis of certain aspects of supply chains such as collaborative inventory management [39], order fulfillment process [10, 36], and requirements analysis [17]. In [9], an integrated agent system which includes both simulated and physical agents is used for supporting real-time online decision making. Physical agents are used for coordinating with other agents to form business processes as well as for detecting the variations occurring in the outside world whereas simulated agents are used for analyzing what-if scenarios to support physical agents in making decisions.

Managing, coordinating, and optimizing supply chain performance by means of multi-agents has also been extensively studied in recent years. In [37], TÆMS agents [16] are equipped with coordination mechanisms to automate and manage a distributed dynamic supply chain. In the manufacturing domain, the Java Agent Development (JADE) Platform (http://jade.tilab.com/) was used to implement an agent-based multi-contract negotiation system for collaborating negotiation in a supply chain network [13]. Furthermore, Yung et al. [33] demonstrate how multi-agent technology and constraint networks can be integrated together to improve the efficiency and transparency of supply chain management. In [33], agents are designed to communicate through the Internet to support dynamic optimization based on Genetic algorithm.

Agent technology has also been applied in online formation of supply chains as well as in the context of Supply Chain Event Management (SCEM). Wang et al. [23] have proposed an agent-mediated approach to on-demand e-business supply chain integration. In their approach, each agent works as a service broker and coordinates their activities across the supply chains which are dynamically set up in response to market requirements. In [23], decision making and coordination among services are modeled as a distributed constraint satisfaction problem whereby solutions and constraints are distributed into a set of services. Each agent then explores its own decisions and coordinates with other agents towards a global solution. In agent-based SCEM [31], a number of agents are defined to monitor event-related information such as disruptions, malfunctions in the operational fulfillment process, and critical orders across the supply chain.

One of the most successful applications of agent technology in simulating supply chain management is the annual Trading Agent Competition in Supply Chain Management (TAC SCM) [12]. The Trading Agent Competition (TAC)[1] [22] is an international forum designed to promote and encourage high quality research into the trading agent problem. The supply chain management game for the trading agent competition (TAC SCM) has been designed jointly by a team of researchers from the e-Supply Chain Management Laboratory at Carnegie Mellon University and the Swedish Institute of Computer Science (SICS) [12]. In TAC SCM, a simulated supply chain management problem is precisely defined for individuals as well as research groups. Based on this uniform framework, researchers are free to design their own agents for competing with other agents in maximizing their profit. This competition provides an excellent opportunity for researchers to implement and test their ideas and designs for various aspects of the supply chain management problem.

The University of Macau team has designed an agent [35] (called UMTac-06) and participated in the 2006 TAC SCM competition. One of the challenges for the University of Macau team in the 2006 TAC SCM competition was to design an agent that can adapt to a fast changing business landscape. Against this background, we implemented a heuristic-based agent and a fuzzy logic-based agent for different stages of the competition. The heuristic-based agent uses a set of rules to control the agent's decision making while the fuzzy logic-based agent employs a fuzzy reasoning mechanism to control the target inventory level. The fuzzy logic-based agent also ad-

[1] http://www.sics.se/tac

justs its own profit margin based on the level of stock holding, market trend, rate of previous successful bidding, and bidding price level of the previous day.

Although a number of approaches (stochastic [19], machine learning [6]) can be applied for adaptively controlling target inventory level and profit margin in supply chain management, we have chosen fuzzy logic [18] for several reasons. Firstly, simplicity in Fuzzy Expert Rules allow supply chain managers to define desired actions and adjustments in layman terms which can be understood and analyzed by other decision makers. As a result, the actions to be taken and factors attributed to such conclusions can be easily verified and tested. Secondly, fuzzy logic represents results or conclusions in degrees rather than in classical binary values. Such type of output is crucial for adjusting subtle responses to the highly dynamic environment. Thirdly, fuzzy systems are well suited for controlling supply chain management processes which are generally ill-defined. These processes are highly interdependent and modeling such activities can be extremely complex.

A brief introduction to the TAC SCM is given in section 2. The heuristic-based agent for qualifying round is described in section 3. The design of the fuzzy logic-based agent is described in section 4. The performance of the agent is discussed in section 5. In section 6 we briefly review related work before summarizing our ideas in section 7.

2. Game overview

In TAC SCM [12], six agents compete in each game. Each game is played for 220 simulated days with each day being 15 seconds long. Each day, agents compete in two markets (supplier side and customer side) in order to maintain their inventory level to build and sell different types of PCs. In a TAC SCM game, 16 types of PCs can be built from four component types: CPUs, motherboards, memories, and hard drives. Each PC type is defined with its constituent components, the number of assembly cycles required, and the market segment (low range, mid range, high range) they belong to. Each component type has two suppliers from the computer hardware manufacturing industry. The suppliers are Pintel for Pintel CPUs, IMD for IMD CPUs, Basus and Macrostar for motherboards, MEC and Queenmax for memories, and Watergate and Mintor for disks. There are eight suppliers in total. The total number of customers is undefined and they are treated as a single entity. An overview of the TAC SCM scenario is given in Figure 1.

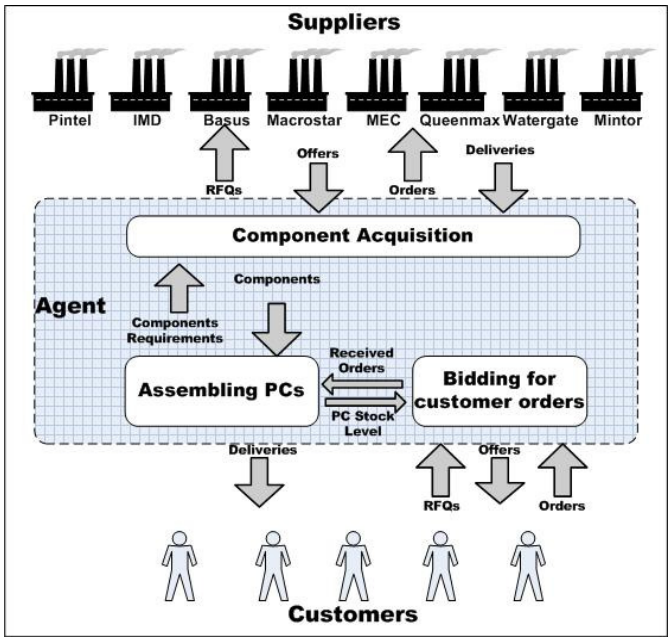

Fig. 1. TAC SCM scenario

Suppliers: Each day an agent is allowed to send a maximum of ten Request For Quotes (RFQs) to each supplier. Selection of an RFQ by a supplier depends on the priority of the agent which is calculated based on the ratio of the number of RFQs and the actual orders. If the supplier can satisfy the order specified in the RFQ in its entirety, an offer is sent as a response. If the supplier cannot supply the entire quantity requested in the RFQ by the requested due date, the supplier will respond by issuing up to two amended offers, each of which relaxes one of the two parameters quantity and due date. All offers made by suppliers are valid for a day and hence require the agent, if interested, to send a confirmation by issuing a purchase order. The production capacity C_d^{ac} of a supplier on a day, *d*, is defined in the TAC SCM specification as follows:

$$C_d^{ac} = \max(1,\ C_{d-1}^{ac} + \text{random}(-0.05, +0.05)\, C^{nom} + 0.01(C^{nom} - C_{d-1}^{ac}))$$

Where C^{nom} denotes the nominal capacity. C_{d-1}^{ac} at the start of the game is typically $C^{nom} \pm 35\%$. At the start of every TAC day the supplier computes C_d^{ac} for each of its production line. The available capacity on day d + i can be defined as:

$$C_{d,i}^{fr} = C_{d,i}^{w} - C_{d,i}^{cm}$$

Where $C_{d,i}^{fr}$ denotes the free capacity for that day. $C_{d,i}^{cm}$ denotes the committed capacity for outstanding agent orders that due to be shipped on day d+i+1. $C_{d,i}^{w}$ denotes the capacity of supplier willing to sell for orders. The offer price of some component c that is due on day d+i+1 is given by:

$$P_{d,i} = P_c^{base}\left(1 - \delta\left(\frac{C_{d,i}^{avl'}}{iC_d^{ac}}\right)\right)$$

where, $P_{d,i}$ is the offer price on day d for an RFQ due on day d+i+1, δ is the price discount factor and have a standard tournament value of 50%. P_c^{base} is the baseline price for components of type C. C_d^{ac} is the supplier's actual capacity on day d. $C_{d,i}^{avl'}$ is the capacity that is available to satisfy some set of requests due on day d+i+1.

Customers: Customers request PCs of different types to be delivered by a certain due date by issuing RFQs to the agents each day. Agents must bid to satisfy the entire order (both quantity and due date) specified in a RFQ. The customer selects the bid with the lowest price (which is less than or equal to the reserve price specified in the RFQ) as the winning bid and the winner will be notified at the start of the next day.

For each RFQ, a penalty is chosen uniformly in the interval of 5% to 15% of the reserve price. Penalties are charged daily when an agent defaults on a promised delivery date. If the agent fails to deliver over a period of five days, the order is cancelled and no further penalties are charged. After the last day of the game all pending orders are charged for the remaining penalty (up to five days) as they can never be delivered.

Agents: Each agent is responsible for controlling the overall process of PC production which includes tasks such as acquiring components, assembling PCs, and delivering to customers. These tasks can be highly influenced by the market situation. Each day, agents issue RFQs to the suppliers. The next day, the suppliers reply to the agents with offers based on their availabilities. Agents then select from these offers (based on the quantities, delivery dates, and prices) and reply to the supplier on the same day. Each day, customers issue requests for quotes of different types of assembled personal computers (PCs) to the agents and, within the same day, the agents reply with offers. Customers then select the best offer based on delivery dates and prices, and reply to the agents on the next day. At the start of each day, each agent receives RFQs for PCs from the customers, and orders won by the agent in response to offers sent to the customers the day before. Also, each agent receives offers for the components in response to the RFQs that the agent had sent to the suppliers the day before. Figure 2 illustrates key daily events involved in running an agent.

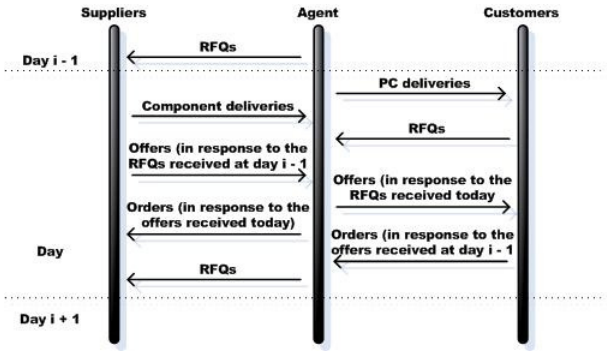

Fig. 2. A TAC SCM day

Each agent is endowed with a PC factory which is capable of assembling any type of PC, and an inventory storing both components and finished PCs. Each day, the agent sends a production schedule to the factory for the production on the next day. The agent also sends a delivery schedule which will cause deliveries to the customer the next day. The supplies (components) can be used for production on the next day after the delivery and PCs are not allowed to be shipped on the day of their production.

Inventory (both finished goods and components) for each agent is charged a storage cost which is a percentage of the base price of components. The storage cost is chosen randomly in a pre-defined range at the start of the game. This cost is applied to the inventory held by the agent at the end of each day.

Agents have accounts in the bank and start the game with no money in the accounts. A fixed interest rate is charged if the balance is in debt or credited if the balance is positive. The storage cost for both finished goods and components is chosen randomly from a predefined range at the start of the game and revealed to all the agents. At the end of the game, the agent with the highest sum of money in the bank is declared as the winner.

In this research, we design two agents: a heuristic-based agent for qualifying round and a fuzzy logic-based agent for seeding round and final round. We briefly summarize both agents' capabilities as follows:

Heuristic-based agent: The agent uses a set of rules to control the agent's decision making in component acquisition and in bidding price setting for customer RFQs. Component acquisition is designed based on the observation that the maximum production capacity of each agent is equivalent to 23 units for each type of PC per day. Heuristic-based agent then orders components to maintain a stock level which is sufficient for assembling 23 PCs per day. For selling those assembled PCs, the agent sets the bidding price for customer RFQs based on the PC inventory. In summary, the heuristic-based agent focuses on maintaining a steady PC inventory level and deploys a conservative form of a buy-to-build strategy [34]. In the buy-to-build strategy, the agent acquires as many components as possible and assembles them into PCs regardless of the orders actually received from the customers.

Fuzzy-based agent: The fuzzy-based agent is designed to overcome some of the issues from the heuristic-based agent. The fundamental architecture of the agent is similar to the heuristic-based agent except the inclusion of fuzzy logic decision making to set *the target inventory level of each type of PC* and *the bidding price of customer RFQs*. The fuzzy-based agent sets the target inventory level of each type of PC based on the market trend and profit margin. The bidding price for the customer RFQs is also set based on the number of PCs in stock, market trend, success rate in customer order bidding, and bidding price level of previous day. In summary, fuzzy-based agent is designed in such a way that it can adapt its decision making behavior based on market situations. This capability is previously unattainable in the heuristic-based agent.

3. Heuristic-based agent

3.1. Component acquisition

First Day: Most of the agents begin ordering components at the early stage of the game. However, since agents have no prior information about the future demand, it is difficult for our agent to choose an appropriate component acquisition strategy. Therefore we design our agent to only order sufficient components for the early period of competition.

From our estimation, the UMTac-06 agent is capable of producing around 23 units of each PC every day. In addition, each type of CPU can be used to assemble 4 types of PC. Other components can be used to assemble 8 types of PCs. Therefore, for 3 days capacity, the agent needs approximately 250 units (23 units x 4 types x 3 days) of CPU and 500 units (23 units x 8 types x 3 days) of other components. Based on this estimation, the agent sends two RFQs to each supplier. The components stated in these RFQs are requested to be delivered on day 3 and day 6. The detail delivery schedule is shown in the first and the second column

of Table 1. The agent also sends 3 additional RFQs to the suppliers for delivering components at day 9, 18, and 25. The components for these RFQs are likely to be cheaper due to their longer delivery due dates.

Table 1

Components ordered at day 1

	Delivery Date				
Component	**Day 3**	**Day 6**	**Day 9**	**Day 18**	**Day 25**
Pintel CPU, 2.0 GHz	250	250	400	300	400
Pintel CPU, 5.0 GHz	250	250	400	300	400
IMD CPU, 2.0 GHz	250	250	400	300	400
IMD CPU, 5.0 GHz	250	250	400	300	400
Pintel Mother-board	500	500	800	600	800
IMD mother-board	500	500	800	600	800
Memory, 1GB	500	500	800	600	800
Memory, 2GB	500	500	800	600	800
Hard disk, 300 GB	500	500	800	600	800
Hard disk, 500 GB	500	500	800	600	800

Day 2 to Day 210: We divide the inventory level of each type of component into three stages: shortage, normal, and maximum stage. For each stage, we define corresponding component ordering plans which are limited by constraints. It is crucial to have constraints in ordering since we need to take into account the components which are yet to be delivered by the suppliers. We then compare the current inventory level against the component stages one by one.

Each agent has 2000 production cycles per day and needs 4 to 7 cycles to assemble a PC. Therefore each agent can produce around 23 units of each type of PC per day. The UMTac-06 agent needs around 92 units of CPU and 184 units of other types of component every day. We define β as the daily required number of each type of component. We set β to 92 units for all CPU components and 182 units for other components (Motherboard, Memory, and Hard Disk). Let V_i be the current inventory level at day i and ECA_i be the expected component arrival at day i. For any simulated TAC SCM day i, the UMTac-06 agent derives the estimated component holding (H_i^n) for next n days as follows:

$$H_i^n = V_i + \sum_{i+1}^{i+n} ECA_i$$

Shortage stage*:* This stage is used to indicate that there will be a potential shortage of components in coming few days. This stage is detected when the estimated component holding for the next 10 days is less than 3β.

$$H_i^{10} < 3\beta$$

If a shortage stage is detected, the agent will send five RFQs to each supplier in which components are requested to be delivered within a short lead time. In our design, each RFQ is defined as a template (RFQ no., order amount, delivery date, constraint). According to TAC SCM game regulations, each agent can send up to five RFQs to each supplier for the products offered by that supplier, for a total of ten RFQs per supplier [12]. In order to maximize the chances of acquiring suppliers' components, the UMTac-06 agent is designed to send following five RFQs to each supplier on every simulated day.

$$\begin{aligned}
&(1, \beta, i+3, No)\\
&(2, \beta, i+5, H_i^5 < 5\beta)\\
&(3, 2\beta, i+10, H_i^{13} < 11\beta)\\
&(4, 2\beta, i+15, H_i^{18} < 14\beta)\\
&(5, 3\beta, i+21, H_i^{24} < 16\beta)
\end{aligned}$$

According to these templates, the agent will send a RFQ to the supplier if it detects that the sum of the current inventory level and components expected to be delivered drops below a certain level. This condition is pre-defined as a constraint. For instance, the UMTac-06 agent will not send RFQ no. 2 to the supplier if estimated component holding for next 5 days is less than 5β ($H_i^5 < 5\beta$). Since RFQs are sent without reserve prices, it is possible that all RFQs could be accepted by the supplier. In such case, the agent could easily exceed its own capacity. With these constraints, we can limit the agent from ordering too many components in short period. In addition, in order to have constant supply of components, RFQs are assigned with different delivery dates which are customized based on the results of our preliminary testing.

Normal stage: The agent is considered to be in the normal stage if the estimated component holding for the next 10 days is less then 8β.

$$H_i^{10} < 8\beta$$

If a normal stage is detected, the agent only orders components for medium term usage by sending following RFQs to each supplier:

$$(1, \beta, i+8, H_i^{9} < 9\beta)$$
$$(2, \beta, i+15, H_i^{18} < 13\beta)$$
$$(3, \beta, i+23, H_i^{25} < 16\beta)$$
$$(4, \beta, i+30, H_i^{34} < 22\beta)$$
$$(5, \beta, i+40, H_i^{43} < 30\beta)$$

Maximum stage: The agent is considered to be in the maximum stage if the estimated component holding for next 14 days is less than 12β.

$$H_i^{14} < 12\beta$$

If a maximum stage is detected, the UMTac-06 agent only orders components for long term production usage by sending following RFQs to each supplier:

$$(1, \beta, i+12, H_i^{14} < 12\beta)$$
$$(2, \beta, i+16, H_i^{18} < 13\beta)$$
$$(3, 2\beta, i+24, H_i^{27} < 20\beta)$$
$$(4, 2\beta, i+35, H_i^{38} < 27\beta)$$
$$(5, 3\beta, i+45, H_i^{48} < 33\beta)$$

After Day 210: To reduce the number of unused components at the end of competition, the UMTac-06 agent stops ordering from suppliers after Day 210.

3.2. Bidding for customer order

Every day, each agent will receive more than 150 RFQs (approx. 1500 PCs) from the customer. The UMTac-06 agent cannot bid all RFQs since it only maintains 5 days production capacity. In addition, an agent is liable to pay heavy penalty if it cannot honor the orders won. Therefore, the UMTac-06 agent assigns higher priority to RFQs which have less number of requested PCs. By choosing such RFQs, the UMTac-06 agent can bid more customer orders and hence the chance of winning is likely to be increased. RFQs with less number of requested PCs have also lower penalty. In addition, customer orders with short delivery dates are more difficult to be fulfilled without a sufficient level of components in the inventory. In contrast, customer orders with long delivery dates could be fulfilled by short to medium term component acquisitions. Therefore, we presume that most of our competitors are less likely to bid RFQs which have short delivery dates. Therefore, in case if the number of requested PCs in two RFQs are the same, the RFQ with shorter due date is given higher priority. However, it is also possible that other agents may have their own capacity reservation mechanisms and could compete for short term customer orders. In addition, fulfilling RFQs with short delivery dates allows our agent to rapidly reduce the inventory and thus costs less storage charge[2].

Day 1 to Day 200: The UMTac-06 agent sets the bidding price (*BP*) according to the total number of PCs in inventory. PCs and components stored in the inventory are subject to storage charge and therefore the agent should avoid having a high inventory level. We define five levels for inventory: *Very Low, Low, Normal, High,* and *Very High.* (see Table 2).

Table 2

Bidding price setting

Level	**Condition**	**Bidding Price (*BP*)**
Very Low	$0 \le I \le 15$	O_{max}
Low	$15 < I \le 50$	O_{avg}
Normal	$50 < I \le 100$	$0.8O_{avg}$
High	$100 < I \le 200$	$1.3O_{min}$
Very High	$I > 200$	O_{min}

where I is the total number of PCs in the inventory and O_{max}, O_{min}, *and* O_{avg} are the maximum, minimum, and average order price of each type of PC from previous day.

After day 200: The UMTac-06 agent tries to sell all PCs from inventory as soon as possible through price reduction. The bidding price after day 200 is defined as follows:

[2] Storage charge is applied to inventory on hand at the end of every simulated day.

$$BP_d = \begin{cases} O_{\min} & 200 < d < 212 \\ O_{\min}\left(1 - \sum_{i=212}^{d} 0.1(i-211)\right) & 212 \le d < 220 \end{cases}$$

where d is the simulated day in the competition.

After analyzing game results from qualifying round, we find that the actual selling price of PCs at the closing time is often higher than those from the early days in the game. One of the possible reasons is that many agents do not have enough components or capacity at the end of the game. From this observation, we decided not to reduce the bidding price for some of the games in qualifying round.

3.3. Assembling PCs

Everyday, the UMTac-06 agent sends a production schedule to its factory. Fulfilling customer orders is crucial for the agent and therefore outstanding orders are given the highest priority. The factory also assembles additional PCs using free capacity whenever possible. From the records of previous testing, we estimate that the UMTac-06 agent can sell around 30 units of each type of PC per day. For each type of PC, the agent calculates the difference between the current stock level and 30 units. The types of PCs to be assembled are then prioritized based on the difference calculated. The types of PC with very low stock level are given highest priority and they are assembled until current stock level is equal to 30 units. The UMTac-06 agent continues to assemble until all remaining free capacity is used.

The agent delivers the assembled PCs to the customers immediately if there are sufficient numbers of units in the inventory. Otherwise, the agent waits until all units are assembled by the factory.

4. Fuzzy-based agent

In the seeding round of the 2006 TAC SCM competition, the UMTac-06 agent deploys fuzzy logic reasoning to control target PCs inventory level and price adjustment for customer orders bidding. In Figure 3, the detail architecture of the fuzzy-based agent is depicted using Gane-Sarson [2] Data Flow Diagram notations.

The UMTac-06 agent is designed with three main components: *Component Acquisition*, *Assembles PCs,* and *Bid for Customer Order*. For each TAC SCM day, the agent sets the target inventory levels of components using fuzzy logic for future acquisition. By using fuzzy logic, the agent calculates the target inventory level based on market trend and profit margin of each type of PC. This situation is depicted in the top and middle area of Figure 3. Agent receives customers' RFQs (Request For Quotes) every day. Before agent sends any offers back to the customers, it sets the bidding price using fuzzy logic by taking into account current PC stock level, market trend, success rate in customer order bidding, and bidding price level of previous day. This situation is depicted in the lower area of Figure 3.

Agent then assembles PCs to fulfill customer orders or to replenish PC stocks. The agent gives highest priority to the outstanding customer orders and extra PCs are assembled if and only if there is any free production capacity.

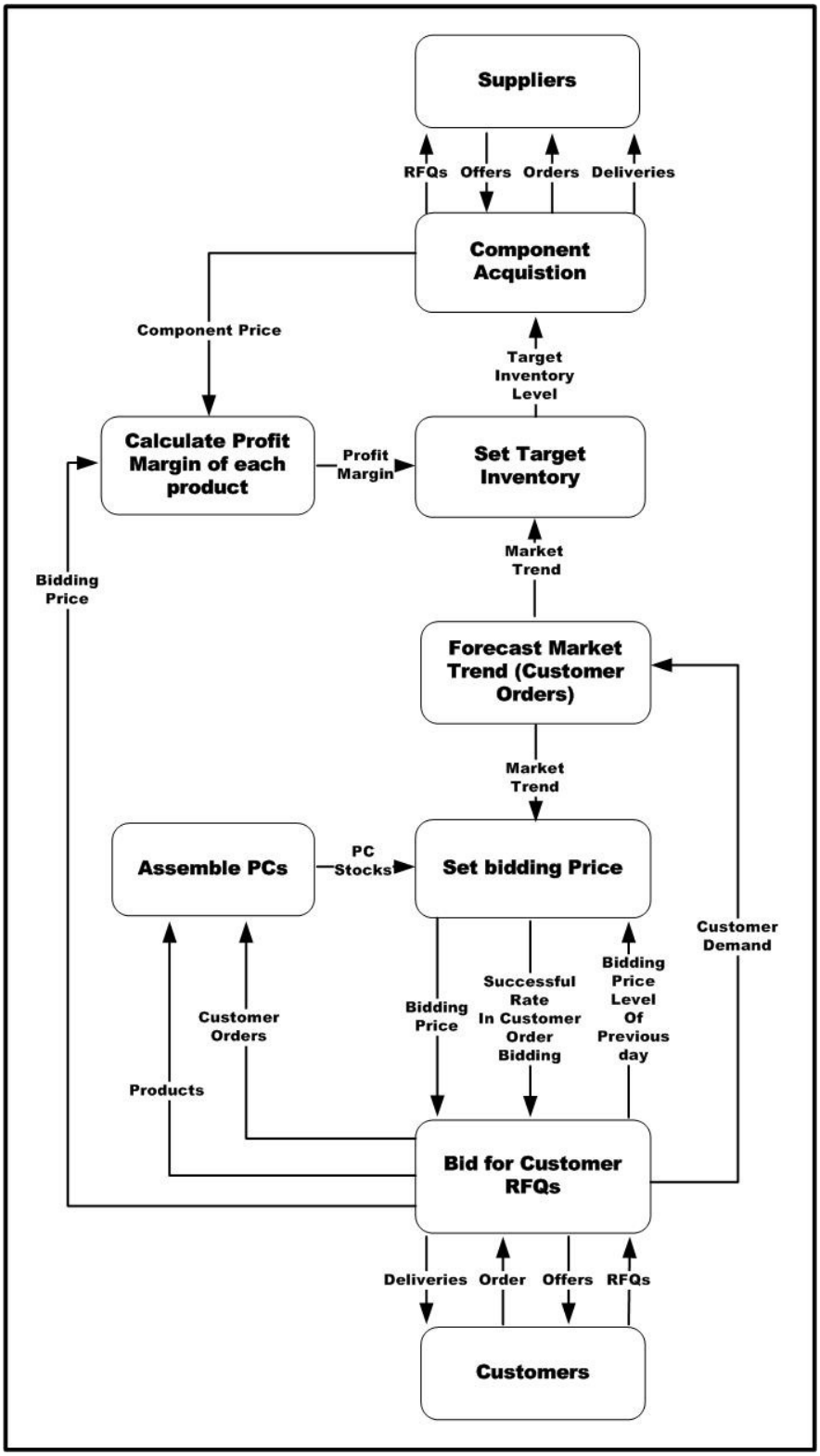

Fig. 3. Fuzzy logic-based agent design

4.1. Component acquisition

First Day: Since the market environments of first day in all games are almost the same, the agent deploys the same component acquisition strategies from qualifying round. (See section 3.1)

Day 2 to Day 210: The UMTac-06 agent uses fuzzy logic to control the inventory level of PCs. The fuzzy reasoning is based on the market trend and previous sale results. There are two steps in calculating target inventory level.

First, we calculate the *average daily customer demand* which is the average number of PCs requested by the customers in their RFQs which are sent to all agents in the competition in the last five days period. Since the UMTac-06 agent may not win all the RFQs, we define the basic daily PC demand of the agent as 20% of the average daily customer demand. Next, we set the basic daily PC demand as the inventory level of the agent. The inventory level is then adjusted with the output from fuzzy logic reasoning. The fuzzy rules take into account two inputs: profit margin and market trend.

1. *Fuzzy input 1, profit margin of each type of PC:* Since profit margin can directly influence the revenue, the UMTac-06 agent is designed to give priority to PCs which have higher profit margins. Based on the records of components the agent has ordered, we calculate the average cost of each type of component. The profit margin is equal to the profit of that type of PC divided by the average cost. Based on the profit margin[3], we define the fuzzy set values as depicted in Figure 4.

2. *Fuzzy input 2, market trend of each type of PC:* Market trend is used to indicate the customer demand of a specific type of PC. The agent is designed to keep higher inventory levels for PCs which have elevated market trend. We use Simple Moving Average (SMA) for calculating market trend. SMA used to indicate whether the demand of PC is likely to increase or decrease in future. The UMTac-06 agent calculates the market trend by dividing 5 days SMA with 10 days SMA. The fuzzy sets for market trend are depicted Figure 5.

[3] In normal situation, the profit margin lies within the range of 15% to 20%. In poor market situation, the profit margin drops to 3% to 8%. In a good market situation, the profit margin can be as high as 25%.

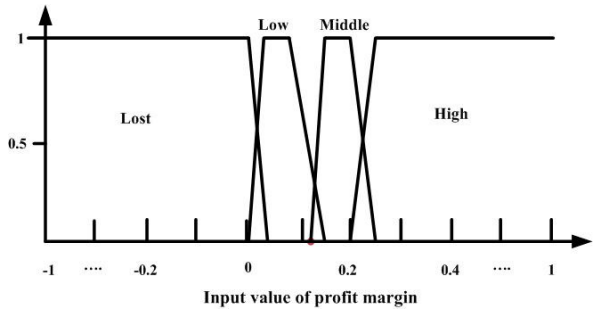

Fig. 4. Fuzzy sets of profit margin for each type of PC

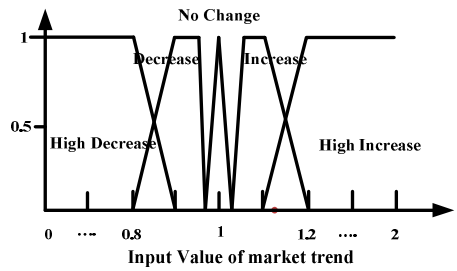

Fig. 5. Fuzzy sets of market trend for each type of PC

Fuzzy output, adjusting target inventory level of each type of PC: We define the fuzzy output as a percentage value which will be added to the basic daily PC demand from step one. We define the fuzzy set value in Table 3 and corresponding fuzzy rules in Table 4.

Table 3

The fuzzy set value of target inventory level

Cut	Reduce More	Reduce	Keep	Add	Add More
-90%, -90%, -90%	-40%, -40%, -25%, -20%	-22%, -18%, -8%, -5%	-4%, 0, 0.4%	5%, 8%, 20%, 22%	20%, 25%, 40%, 40%

Based on these fuzzy variables, 20 rules can be generated from all possible combination of fuzzy input and output. Examples of generated fuzzy rules are given in Table 4.

Profit Margin (P) $\in$ *{lost, low, middle, high}*
Market Trend (MT) $\in$ *{high decrease, decrease, no change, increase, high increase}*
Adjusting Target Inventory Level (T) $\in$ *{cut, reduce more, reduce, keep, add, add more}*

Table 4

The rule set for target inventory level

IF P is lost and MT is decrease THEN T is cut.
IF P is lost and MT is increase THEN T is reduce.
IF P is low and MT is decrease THEN T is reduce.
IF P is low and MT is increase THEN T is keep.
IF P is middle and MT is decrease THEN T is keep.
IF P is middle and MT is increase THEN T is add.
IF P is high and MT is decrease THEN T is keep.
IF P is high and MT is increase THEN T is add more.

According to rule 1 in Table 4, if the profit of one type of PC is considered to be lost and the market trend for that type of PC is quickly decreasing, the agent should cut substantial amount of that type of PC.

The agent then orders components from the suppliers based on the target inventory level calculated. Suppose that a CPU with 2.0GHz can be used to assemble either a PC of type A or B. If the target inventory level of type A is 30 units and type B is 40 units, then the required number of CPU with 2.0GHz will be equivalent to 70 units. We define this value (e.g. 70 units) as the daily required number of a particular component (abbreviated as μ). Similar to the heuristic-based agent, we divide the inventory level of each type of component into four stages: *critical, shortage, normal* and *maximum.*

Critical stage: This stage is used to indicate that there will be a severe shortage of components in a short term. The agent is considered to be in the critical stage if the estimated component holding for the next 5 days is less then 4μ.

$$H_i^5 < 4\mu$$

If a critical stage is detected, the UMTac-06 agent only orders components for short term production usage by sending following four RFQs to the suppliers. The fifth RFQ is used for probing price.

$$(1, \mu, i+5, H_i^9 < 6\mu)$$
$$(2, \mu, i+10, H_i^{12} < 10\mu)$$
$$(3, 2\mu, i+15, H_i^{18} < 15\mu)$$
$$(4, 2\mu, i+20, H_i^{24} < 18\mu)$$

Shortage stage: This stage is used to indicate that there will be a potential shortage of components in a short term. The agent is considered to be in the shortage stage if the estimated component holding for the next 9 days is less then 6μ.

$$H_i^9 < 6\mu$$

If a shortage stage is detected, the UMTac-06 agent only orders components for medium term production usage by sending following RFQs to the suppliers:

$$(1, 2\mu, i+10, H_i^{12} < 11\mu)$$
$$(2, 2\mu, i+15, H_i^{18} < 15\mu)$$
$$(3, 2\mu, i+20, H_i^{24} < 19\mu)$$
$$(4, \mu, i+25, H_i^{30} < 25\mu)$$

Normal stage: The agent is considered to be in the normal stage if the estimated component holding for the next 14 days is less then 10μ.

$$H_i^{14} < 10\mu$$

If a normal stage is detected, the UMTac-06 agent only orders components for long term production usage by sending following RFQs to the suppliers:

$$(1, 2\mu, i+15, H_i^{20} < 14\mu)$$
$$(2, \mu, i+20, H_i^{26} < 18\mu)$$
$$(3, \mu, i+25, H_i^{30} < 21\mu)$$
$$(4, \mu, i+35, H_i^{40} < 28\mu)$$

Maximum stage: This stage is used to indicate that available components in the inventory are estimated to be sufficient for medium term requirement. The agent is considered to be in the maximum stage if the estimated component holding for the next 14 days is greater than or equal to 10μ.

$$H_i^{14} \geq 10\mu$$

If a maximum stage is detected, the UMTac-06 agent will only order a small quantity of components for future usage.

$$(1, 2\mu, i+20, H_i^{20} < 14\mu)$$
$$(2, \mu, i+25, H_i^{26} < 18\mu)$$
$$\left(3, \frac{1}{2}\mu, i+30, H_i^{35} < 22\mu\right)$$
$$\left(4, \frac{1}{2}\mu, i+35, H_i^{40} < 26\mu\right)$$

Day 210 to 220: After day 210, the UMTac-06 agent stops ordering from suppliers in order to reduce the number of unused components.

4.2. Bidding for customers orders

Selection of customer RFQs: The UMTac-06 agent is designed to select RFQs which are likely to yield high profit. The agent selects RFQs based on four criteria: (1) due date, (2) reserve price, (3) the number of units, and (4) penalty.

- *Due date:* Due dates of customers' RFQs vary from 3 to 12 days [2]. The UMTac-06 agent assigns higher priority to RFQs with shorter due date since we predict that the other agents may not have enough PCs or free capacity to fulfill these RFQs. We define the priority PD based on the delivery date as follows:

  $P_D(x) = 1000 - 100(x - 3)$,
  where x is the required delivery date and $12 \geq x \geq 3$.

- *Reserve price:* The UMTac-06 agent assigns higher priority to RFQs which impose high reserve price. Reserve price is usually defined as 75% to 125% of the nominal price of a PC [2]. We define the priority P_R for reserve price as follows:

  $P_R(x) = 25(x - 75) + 25$,
  where x is the reserve price (defined as the percentage of the nominal price [2]) and $125 \geq x \geq 75$.

- *The number of units:* The number of units requested in any customer RFQ is within 1 to 20 [2]. All agents have limited production capacity. Therefore, if the number of units requested in a RFQ is low, the agent can bid more RFQs and hence increase the chance of winning more customer orders. In addition, if the agent cannot fulfill the RFQ which contains low number of requested units, the penalty is also likely to be low as well. We define the priority P_U for different number of units as follows:

  $P_U(x) = 2000 - 100(x - 1)$,
  where x is the number of requested units in a RFQ and $20 \geq x \geq 1$.

- *Penalty:* The range of penalty for a customer RFQ is betweens 5% to 15% of the customer reserve price per day [2]. The agent assigns higher priority to RFQs which impose lower penalty. We define the priority PN based on the penalty as follows:

  $P_N(x) = 50(x - 5) + 50$,
  where x is the penalty and $15 \geq x \geq 5$.

After the agent has received all customer RFQs, it calculates the priority of each RFQ as follows:

$$Priority = P_D + P_R + P_U + P_N$$

In these formulas, P_i ($i \in \{D, R, V, N\}$) is not normalized and the range of Pi value is used to reflect the priority assigned to each criteria. In order to maximize the chance of winning more customer orders, the UMTac-06 agent assigns highest priority to RFQs with less number of units. The UMTac-06 agent only bids for customer orders according to its available capacity. We presume that the penalty incurred could be lower due to this conservative strategy. Therefore, the agent assigns lowest priority to RFQs with lower penalty. The agent assigns medium priorities to RFQs with shorter due dates as well as to RFQs with high reserve price. Finally, the RFQs are ordered from highest to the lowest priority for bidding. The agent also makes sure that it only bids for the RFQs which can be fulfilled by the current inventory (including yet to be delivered components).

Bidding price setting: After the agent has selected the RFQs based to their priority, it determines the bidding price based on the number of available PCs in the stock, market trend, recorded success rate, and bidding price level of previous day. The agent uses fuzzy logic reasoning to determine the price adjustment for bidding price setting.

- *Fuzzy input 1, number of PCs in stock:* Every agent maintains an inventory of assembled PCs. Determining the level of current stock based on the number of available PCs in the inventory is a difficult task since the notion of high or low correlates to different market scenario. Therefore, the UMTac-06 agent calculates the PC inventory level based the value of μ which reflects the current market environment. The PCs inventory level is divided into *very low, low, average, high,* and *very high*. The fuzzy set value and corresponding fuzzy sets are given in Figure 6.

- *Fuzzy input 2, market trend of each type of PC:* Market trend is calculated in the same way as described in section 4.1. Fuzzy sets for market trend are depicted in Figure 7.

- *Fuzzy input 3, success rate in customer order bidding:* Low success rate in winning customer orders may indicate certain problems in the bidding price setting mechanism. One possible reason for having low success rate is that the agent's bidding price (selling price) could be too high to be acceptable by the customers. In that case, the agent should reduce the bidding price. In contrast, if the success rate in winning customer orders is too high, the agent should increase the bidding price to secure more profit. To reflect these situations, success rate is defined as the total number of PCs which are actually ordered by the customers divided by the total number of PCs bided in the previous day. Fuzzy sets for the success rate in customer order bidding are depicted in Figure 8.

- *Fuzzy input 4, bidding price level of previous day:* The bidding price level of previous day is defined as the bidding price of that particular type of PC divided by the nominal price which is defined in [12]. Fuzzy sets for bidding price level of previous day are depicted in Figure 9.

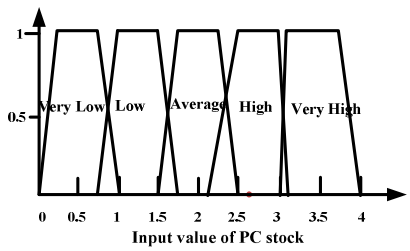

Fig. 6. Fuzzy sets for the number of PCs in stock

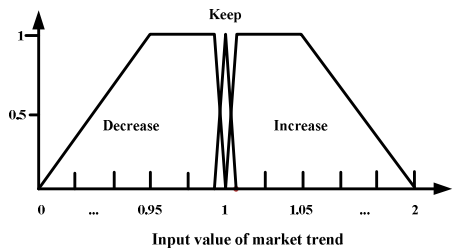

Fig. 7. Fuzzy sets of market trend for each type of PC

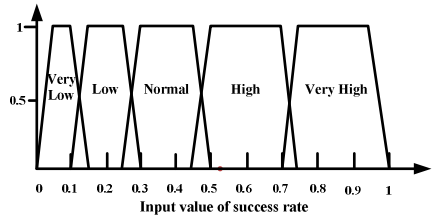

Fig. 8. Fuzzy sets for the success rate in customers' order bidding

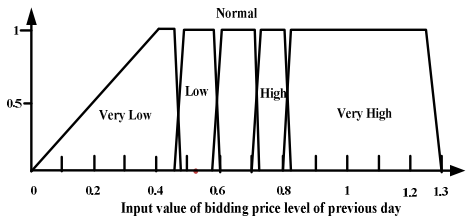

Fig. 9. Fuzzy sets for bidding price level of previous day

- *Fuzzy output, price adjustment:* The output of fuzzy logic reasoning is the price adjustment (in percent) which is to be applied to the bidding price of previous day. The fuzzy set for price adjustment is given in Table 5.

Table 5

The fuzzy set of price adjustment

Reduce More	Reduce Little	Reduce	Keep	Add	Add Little	Add More
-17%, -16%, -12%, -11%	-12%, -11%, -8%, -7%	-8%, -7%, -3%, -2%	-2%, 0, 2%	2%, 3%, 7%, 8%	7%, 8%, 11% 12%	11%, 12%, 16%, 17%

From all possible combination of fuzzy input and output, 375 rules can be generated. After simplification, we obtain 162 fuzzy rules. Examples of several rules from the simplified set are given in Table 6.

PCs in stock (PS) $\in$ *{very low, low, average, high, very high}*

Market Trend (MT) $\in$ *{decrease, keep, increase}*

Successful rate in customer order bidding (SR) $\in$ *{very low, low, normal, high, very high }*
Bidding price level of previous day (BP) $\in$ *{very low, low, normal, high, very high}*
Price adjustment (PA) $\in$ *{reduce more, reduce little, reduce, keep, add, add little, add more}*

Table 6

Example of fuzzy rules after simplification

IF PS is very low, SR is low THEN PA is keep.
IF PS is very low, SR is normal THEN PA is add.
IF PS is very low, SR is high THEN PA is add more.
IF PS is low, SR is low THEN PA is reduce.
IF PS is low, SR is normal THEN PA is keep.
IF PS is low, SR is high THEN PA is add.
IF PS is average, SR is low THEN PA is keep.
IF PS is average, SR is normal THEN PA is Add.
IF PS is average, SR is high THEN PA is Add more.
IF PS is high, SR is low THEN PA is reduce.
IF PS is high, SR is normal THEN PA is reduce.
IF PS is high, SR is high THEN PA is add.
IF PS is very high, SR is low THEN PA is reduce more.
IF PS is very high, SR is normal THEN PA is reduce.
IF PS is very high, SR is high THEN PA is add.

According to rule 1 in Table 6, if the number of PCs in stock is very low and success rate in customer order bidding is low, the price adjustment should be set to "keep" (i.e. only requires minimum adjustment).

4.3. Assembling PCs

The UMTac-06 agent assembles PCs based the target inventory level calculated in section 4.1. For each type of PC, the agent calculates the difference between the target inventory level and the current stock level. The types of PC to be assembled are then prioritized based on the difference calculated. The types of PC with very low stock level are given highest priority and are assembled until current stock level is equal to the target inventory level. If there are free factory cycles, the agent will continue assembling more PCs until the stock level is equal to four days requirement (4μ). The agent delivers the assembled PCs to the customer without delay if there are sufficient numbers of units in the inventory. Otherwise, the agent waits until all units are assembled by the factory.

4.4. Agent for final round

For the final round of TAC SCM competition, we have revised the fuzzy logic-based agent with some additional heuristic rules.

- *Bidding price for customer RFQs:* In the previous rounds, the agent only selects a RFQ if the reserve price is higher than the calculated bidding price. In the final round, in order to increase the revenue and to secure more customers orders, the agent is revised to enhance its flexibility in bidding price setting. In the revised version, the UMTac-06 agent scans customer RFQs three times. In the first round, the agent uses the bidding price calculated from fuzzy logic-based price adjustment to select RFQs from the pool. In the second round, the agent scans the remaining RFQs from the pool and selects the RFQs using 2% reduced bidding price. In the third round, the agent will again scans the remaining RFQs from second round and selects RFQs using 4% reduced bidding price.
- *Components ordering:* In the seeding round, the UMTac-06 agent uses fuzzy logic to control the target PCs inventory level. However, the agent only takes into account the number of units in each RFQ sent by the customer. It does not consider the total number of units actually being sold. In fact, the agent should acquire more components if it can sell a large number of PCs. To correct this problem, we apply following additional heuristic rules:
  - If the total number of units ordered by the customer is greater than 350 units in a TAC day, the agent will increase the component order by 4%.
  - If the total number of units ordered by the customer is in the range of 300 to 350 units, the agent will increase the component order by 2%.
- *Reserve price for components:* In the final round, the UMTac-06 agent is revised to set the reserve price for components ordering based on the profit margin of the PC. By doing so, it ensures that the agent will not sell PCs if it does not make any profit. To calculate the reserve price, the profit margin is added to the average price of that component.
- *Balancing stock level of components:* To assemble a PC, the agent requires four types of component. Therefore, if one type of component is unavailable or out of stock, the agent can not assemble that particular type of PC. In the seeding round, the UMTac-06 agent is designed not to order any component after day 210 to avoid com-

ponent build up. In the final round, after day 210, if one type of component is less than the average component inventory level, the agent will send RFQs to the suppliers to increase the stock level of that particular component. We set the average component inventory as 920 units which is the maximum usage capacity of the agent for 10 remaining days.

5. Performance comparison

We analyze the performance of the heuristic-based agent and the fuzzy logic-based agent from qualifying, seeding, quarter finals, semi finals, and second finals rounds of the 2006 Trading Agent Competition for Supply Chain Management.

5.1. Heuristic-based agent Vs fuzzy-based agent

Factory utilization: We find that the adaptability of the UMTac-06 agent has increased after fuzzy logic-based inventory level control is deployed. During the qualifying round, the heuristic-based agent maintained similar factory utilization regardless of market environments. Specifically, the agent kept high factory utilization in any market environment. After we have deployed fuzzy logic-based inventory control, we find that the UMTac-06 agent adaptively reduces the production for some of the non-profitable PCs according to the market environment.

In Figure 10, we compare the factory utilization levels of both agents in different market environments. We classify the market situation into three categories: *poor*, *general*, and *favorable*. A game is defined as a poor market if three or more agents get negative result (profit). A game is defined as a favorable market if less than three agents get negative result. In general case, we take into account all games regardless of their results. During the qualifying round, factory utilization in favorable market is 80% and in poor market is 77%. The small difference in these two markets (just 3%) indicates that the heuristic-based agent does not adapt well to the market environment. However, in the seeding and final rounds, factory utilization for favorable market is 67% and for poor market is 58%. The difference in these two markets has increased to 9%. It shows that our fuzzy logic-based agent is capable of adapting its production level with respect to the market environment.

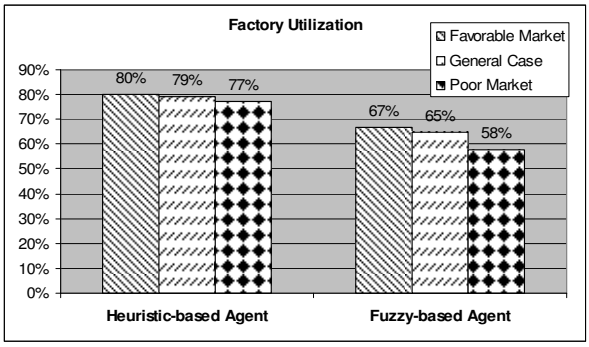

Fig. 10. Factory utilization levels for qualifying round (the heuristic-based agent) and seeding/final round (the fuzzy-based agent)

Average PC selling price: We have extracted the average selling price for each type of PC in all rounds (see Figure 11). We can notice that the average selling price of fuzzy-based agent from seeding and final rounds is significantly higher than those of heuristic-based agent from qualifying rounds. It shows that the adaptability of agent has increased after fuzzy logic-based strategy for bidding price setting in customer orders has been deployed.

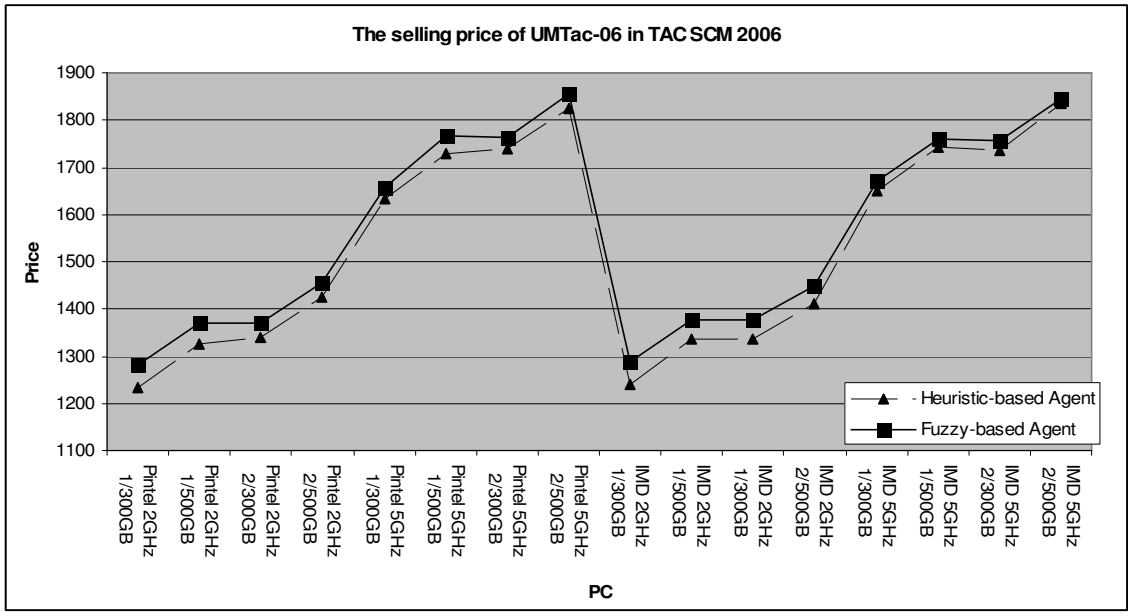

Fig. 11. Average PC selling price

Success rate in bidding customer order: We have also observed that the success rate in bidding for customer orders has also increased significantly in seeding and final rounds (28.33% for the heuristic-based agent, 31% for the fuzzy logic-based agent) after customer RFQs selection strategy and fuzzy logic-based price setting have been deployed.

Overall performance: Four factors are used to measure the overall performance of UMTac-06. In Figure 12, we can observe that the overall profit is significantly increased in seeding round after a fuzzy logic-based strategy has been deployed. In addition, the interest charged, storage cost and penalty are also decreased. It also shows that the agent's strategy on just-in-time production and delivery has effectively reduced the overhead cost.

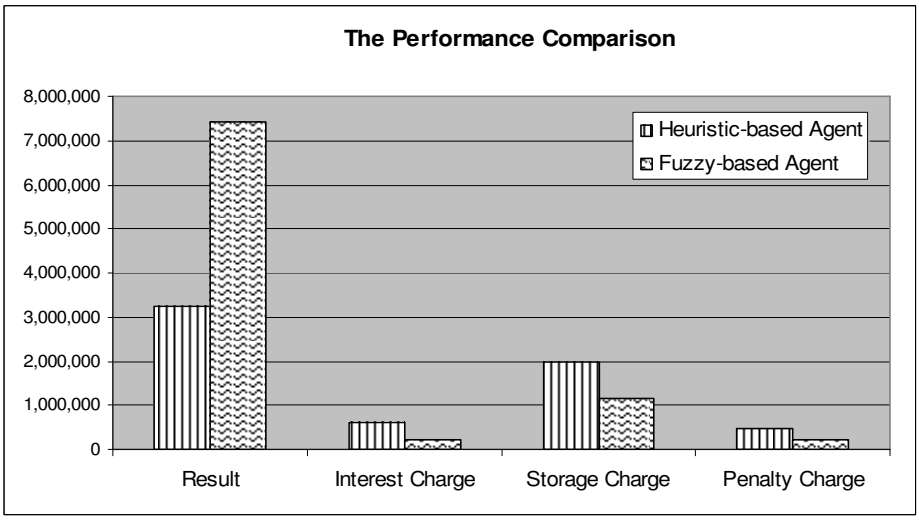

Fig. 12. Performance comparison

5.2. Comparison with top three agents

Storage cost: The fuzzy-based UMTac-06 agent has competed in seeding round (96 games), quarter finals round (9 games), semi finals round (16 games), and second finals round (16 games). From the game results, we calculate the average storage cost of UMTac-06 agent and the other agents. We find that the average storage cost of UMTac-06 is lower than the average cost of other agents (see Figure 13).

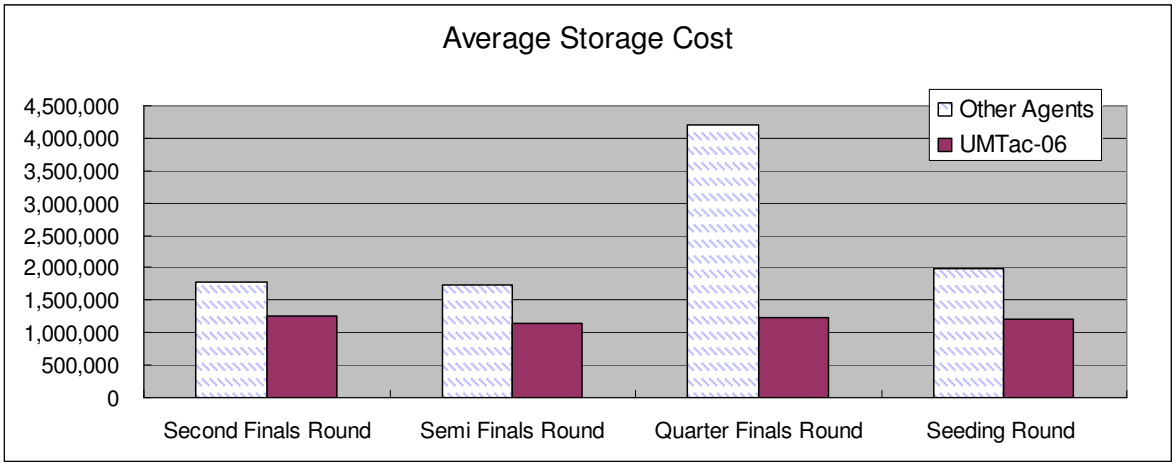

Fig. 13. Average storage cost

In Figure 14, we compare the average storage cost of fuzzy-based UMTac-06 agent with top three agents (TacTex, PhantAgent, and DeepMaize) from 2006 competition. The result shows that our agent has relatively low storage cost in all rounds.

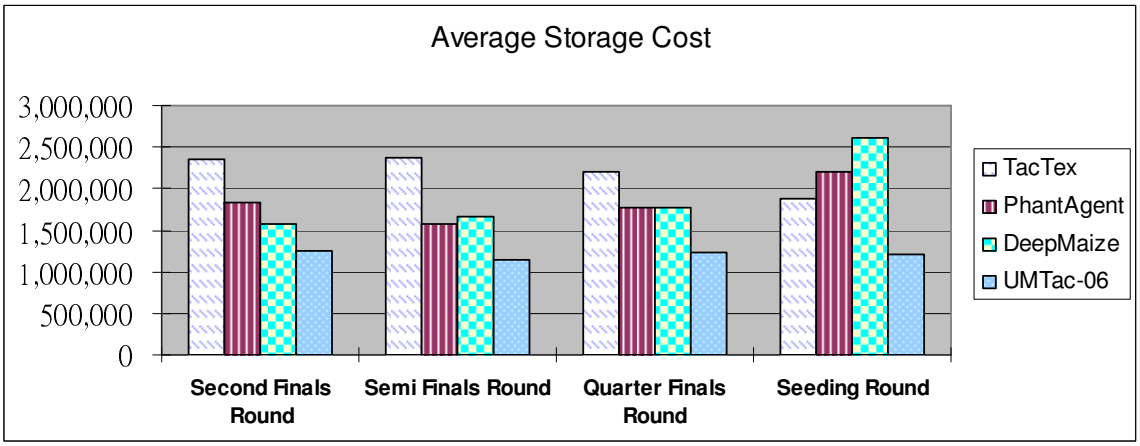

Fig. 14. Comparison of average storage cost with top three agents from 2006 competition

Penalty: In Figure 15, we compare the average penalty of fuzzy-based UMTac-06 agent with top three agents. Due to our agent' conservative bidding strategy for customer orders, we have relatively less penalty compared to other agents. However, our agent had high penalty in second finals round due to the network problem during game 8637.

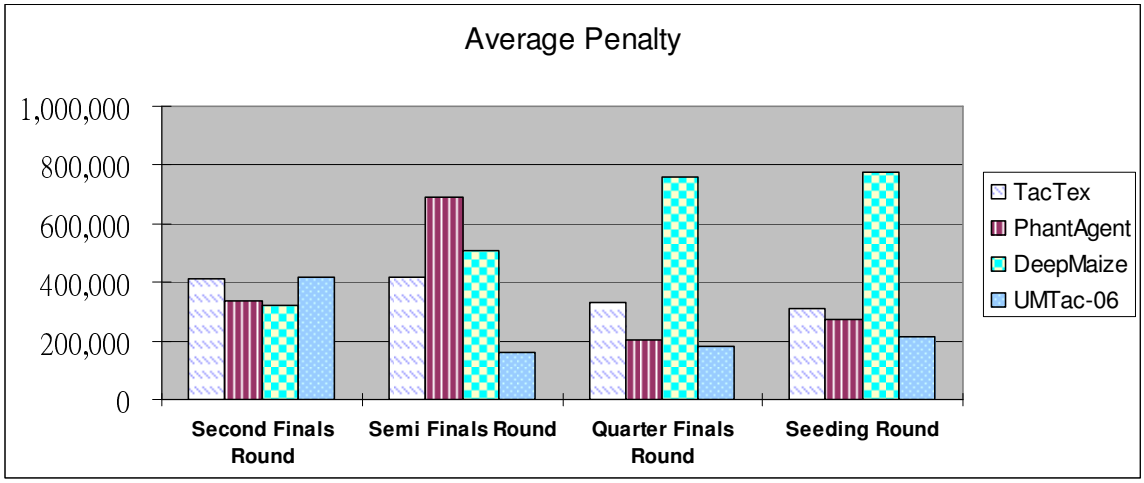

Fig. 15. Comparison of average penalty with top three agents form 2006 competition

Customer orders: In Figure 16, we compare the average number customer orders received by the fuzzy-based UMTac-06 agent with top three agents from the competition. We find that our agent has won less customer orders in second finals, semi finals, and quarter finals Round. One noticeable reason for resulting low successful rate in winning customer order is that the UMTac-06 agent lacks a prediction mechanism for bidding customer orders.

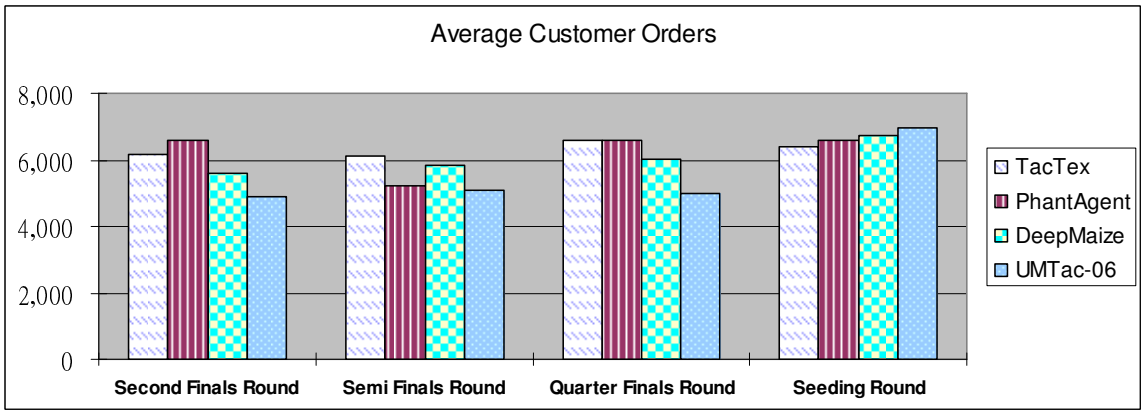

Fig. 16. Comparison of average number of customer orders with top three agents form 2006 competition.

5.3. Overall performance

The UMTac-06 agent has competed against 23 international competitors in 2006 Trading Agent Competition. The UMTac-06 agent based on heuristic rules has scored 17th position in qualifying round[4]. In the seeding round, the UMTac-06 agent based on fuzzy logic has secured 8th position[5]. After the seeding round, the fuzzy logic-based UMTac-06 agent is modified with additional heuristics rules. The revised agent has secured second position in his group in quarter finals and 13 agents are eliminated. In the second finals round, the UMTac-06 agent has secured 4th position[6]. The average score of UMTac-06 agent in 2006 TAC SCM competition is depicted in Figure 17, 18, 19, and 20.

[4] http://www.sics.se/tac/page.php?id=57

[5] http://www.sics.se/tac/page.php?id=57

[6] http://www.sics.se/tac/page.php?id=59

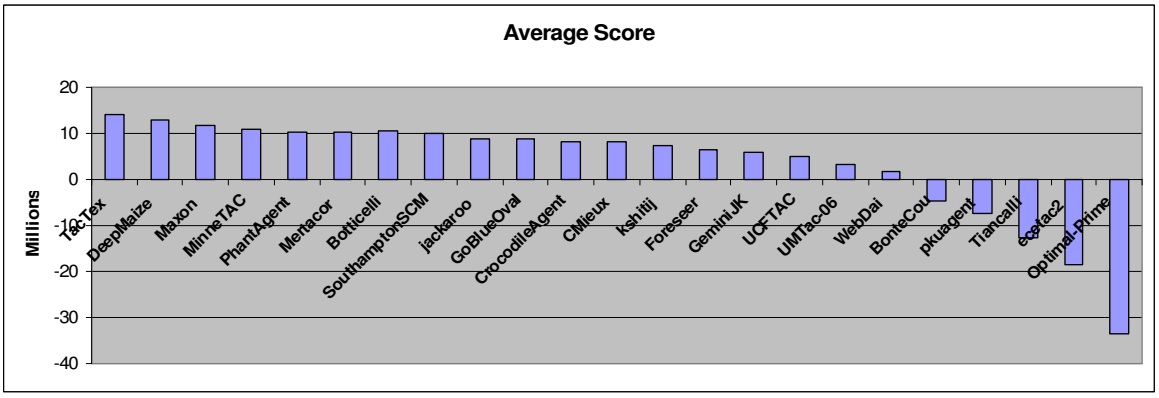

Fig. 17. The qualifying round in 2006 TAC SCM competition

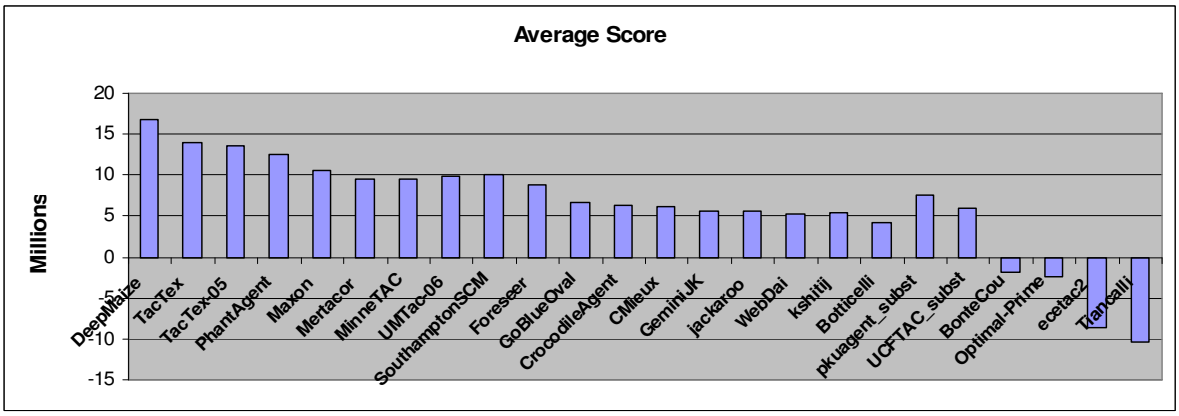

Fig. 18. The seeding round in 2006 TAC SCM competition

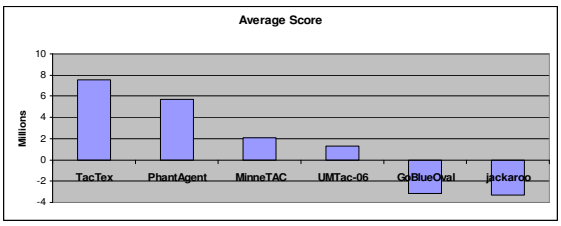

Fig. 19. The semi finals round for Group 2 in 2006 TAC SCM competition

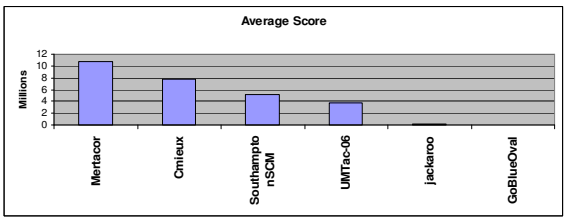

Fig. 20. The second finals round in 2006 TAC SCM competition

6. Related work

In the 2003 and 2004 TAC SCM competitions, the majority of the agents adopted make-to-plan and make-to-order (also known as buy-to-build and build-to-order [29]) as their main strategies. The results of an empirical comparison of these two basic types of strategies, in the light of their application to the TAC SCM competition, have been reported in [40]. The result of the analysis reveals the strengths and weaknesses of both strategies in various aspects. The UMTac-06 agent was partly based on the overall results of the simulation analysis from [40]. The main objectives of fuzzy reasoning in UMTac-06 agent are (a) to adjust the target inventory level for component acquisition and (b) to set a suitable bidding price for customer orders. In recent TAC SCM competitions, researchers have developed a variety of methods for controlling inventory level and setting bidding price for customer orders. From a large number of TAC SCM agent reports, we have selected to review some agents which are similar to our agent design.

6.1. Component acquisition

A similar strategy in component acquisition based on a threshold value was adopted by TacTex-06 agent [5]. TacTex-06 keeps the inventory of each component above a certain threshold (determined experimentally). The agent then determines the deliveries that will be needed to maintain the threshold on each day in the future. The agent then adds these deliveries to the actual deliveries expected from previously placed component orders. A more aggressive approach in component acquisition was adopted by the Tiancalli06 agent [4], which calculates daily inventory levels based on components acquired/sold and existing assembled PCs. The Tiancalli06 agent then bulk purchases at least 35% of the required components on the first day. The reserve prices for the component orders after the first day are then adjusted by heuristic rules.

CrocodileAgent [1] from 2005 TAC SCM competition is also designed to order a large number of components on day-0 by sending RFQs with the delivery dates: 3, 9, 17, 27 and 69 and the reserve prices are set to 102%, 107%, 92%, 82% and 77% of the nominal price on the respective delivery dates. The agent also ordered additional components throughout the game by taking into account the current inventory and the number of components that have been ordered. CrocodileAgent's strategy is similar to UMTac-06 agent's 0 day component acquisition heuristics except that our agent does not set any reserve price for the RFQs in the qualifying and seeding rounds.

A more sophisticated approach for component acquisition was adopted by the Deep Maize [3] agent from the 2005 TAC SCM competition. Deep Maize estimates component values from the projected production schedule and then creates RFQs for each future day that maximizes the total value of components purchased less the predicted price for the given day. These RFQs are heuristically arranged into five requests per supplier per product line. For acquiring components, the agent maintains a projection of the cumulative number of components necessary for each future day by taking into account current inventory, expected future deliveries of components, and components scheduled for production. The main difference between UMTac-06 agent and Deep Maize's design is that Deep Maize acquires components according to marginal profit per factory cycle which is the difference between the expected marginal revenue and marginal replacement costs (including storage charge and interest rates), whereas in UMTac-06 components are acquired based on profit margin which is calculated from the selling price and the average cost of the PC.

6.2. Inventory management

An inventory control strategy based on thresholds was deployed by Mertacor agent [11] in 2005 TAC SCM competition. The agent first determined inventory levels (thresholds) that need to be satisfied. These thresholds are calculated in real time for each component using the demand for a specific component, supplier lead time, and a safety factor denoting the service level. Mertacor orders components in two phases. On day 0 and day 1, Mertacor orders a large number of components. Starting from day 2, the agent sends RFQs regularly to the suppliers based on normal, critical, and early procurement decisions. The HarTac agent [30] also deploys a similar approach for controlling inventory levels by maintaining a reasonable quantity of components in stock at all time. The HarTac agent defines three component thresholds: Critical, Minimum and Maximum. The critical level for CPU components is 240 units (for non-CPU components it is 120 units) and the minimum level for CPU components is 1600 (for non-CPU components it is 800 units). HarTac defines different maximum levels for each game and the value is reduced as the

game progresses to the end. The main difference between our approach and HarTac's design is that UMTac-06 controls the inventory level based on the *daily required number of a particular component (μ)* which varies with the target inventory level throughout the game whereas in HarTac, critical and minimum level are defined as constant thresholds.

A probabilistic approach to inventory management was implemented in the NaRC agent [32] from the 2004 TAC SCM competition. The agent deployed a Markov Decision Process (MDP) to optimize the level of inventory each day, while lowering total cost. The agent then used Dynamic programming to determine the optimal action at each state in the MDP. These actions include submitting RFQs to the various suppliers and accepting/rejecting quotes.

An innovative inventory management approach was deployed in RedAgent [28] from the 2003 TAC SCM competition. The agent deployed an internal market which was designed with a number of simple heuristic-based agents. RedAgent allocates one internal market (a variation of sequential, sealed-bid double auctions) for each type of PC, each component type, and for the production cycles. The main difference between our approach and RedAgent's design is that the UMTac-06 agent is designed to take into account the profit margin and market trend in adjusting target inventory level, whereas in RedAgent the target inventory level is set to the number of PCs to be assembled for 10 days of operation which is limited by a lower and upper bound value.

6.3. Bidding for customer orders

The TacTex-06 [5] agent from the 2006 TAC SCM competition identified the set of bids in response to customer RFQs that will maximize the expected profit based on the remaining production resources for the next 10 days. For each RFQ, the agent computes both the probability of winning and the expected profit as a function of price. The agent then chooses RFQs based on a greedy production scheduler for bidding. The main difference between our approach and TacTex-06's design is that UMTac-06 is designed to take into account the level of stock holding, market trend, rate of previous successful bidding, and bidding price level of previous day for calculating the bidding price for customer orders, whereas in TacTex-06, the probability of winning and the expected profit are used to determine the bidding price.

A heuristic-based approach in selecting customer orders and calculating price adjustment was adopted by the Tiancalli06 [4] agent from the 2006 TAC SCM competition. In their agent design, customer orders are selected based on a discount factor and the cost the PCs. The Tiancalli06 agent then selects a customer order if the price that the customer is willing to pay is greater than or equal to the cost of the product after a discount factor is applied. The discount factor is calculated from the ratio of successful customer orders. The main difference between our approach and Tiancalli06's design is that the Tiancalli06 agent uses the cost of the PCs as a base price whereas in the UMTac-06 agent, the price adjustment is added to the bidding price of the previous day. Therefore, in a poor market situation the UMTac-06 agent may bid customer orders at a loss.

The UMTac-06 agent adopts similar strategies deployed by CrocodileAgent [1] and RedAgent [28] in customer order bidding. CrocodileAgent [1] prioritizes customer RFQs according to the profit. The agent's desired profit is defined as a percentage of the basic PC price and it is adjusted depending on customer PC demand, factory utilization, previous prices and other factors. RFQs are then sorted in chronological order of their delivery dates. Likewise, RedAgent [28] also computes offers to the customers by taking a running average of the closing price from the PC market and adding a margin for each necessary production cycle. The margin is adjusted based on the number of available production cycles.

In the PackaTac agent [8], customer orders are prioritized according to the inventory level. The agent then sets the bidding price based on profit margin. In a high demand market, the agent sets the profit margin higher, and in a low demand market the agent sets the profit margin lower. In addition, the profit margin is adjusted if the agent won too many or too few orders. The main difference between the UMTac-06 agent and PackaTac's design is that our agent only selects the RFQs for bidding based on due date, reserve price, the number of units, and the penalty.

An interesting approach in customer order bidding is adopted by Mertacor [11]. The agent uses a statistical model to predict the winning price of a customer order. It then decides which customers' RFQs to bid on based on the anticipated profit. In contrast to Mertacor, the UMTac-06 agent lacks a prediction mechanism for bidding on customer orders.

6.4. Agents with fuzzy logic reasoning

The most relevant work to our own is the SouthamptomSCM agent [21] from the 2004 competition.

In [21], fuzzy logic reasoning was employed for setting bidding prices for customer orders. The difference between SouthamptonSCM agent and the UMTac-06 agent is that in [21] the agent sets its bidding prices based on market demand, its own inventory level, and the time into the game, whereas in our approach, the UMTac-06 agent takes into account the level of stock holding, market trend, rate of previous successful bidding, and bidding price level of the previous day. In addition, our agent is designed to adjust its target inventory level based on profit margin and market trend.

Fuzzy logic was also used in one of the agents designed for a different competition scenario: Trading Agent Competition for Travel Industry (TAC Classic). SouthamptonTAC [20] is an adaptive agent designed for TAC Classic Competition (Travel Agents). The agent predicts clearing prices of auctions for hotel by using fuzzy reasoning techniques. SouthamptonTAC estimates the auction closing prices by using fuzzy reasoning methods. It also uses multiple rule-bases to make its predictions estimates [20].

In addition to these agents' design from TAC SCM competitions, the use of fuzzy logic reasoning for modeling supply chain management has also been extensively studied by a number of researchers. Wang et al. [15] have developed a fuzzy decision method for handling uncertainties (such as customer demand, supplier's delay time, material transit time and production time) and for determining inventory strategies. In modeling supply chains, Petrovic et al. [7] have also used fuzzy modeling techniques to determine the order quantities for each inventory based on customer demand, supply deliveries and external or market supply. In their approach, uncertainties are described by vague and imprecise phrases that are interpreted and represented by fuzzy sets. Fuzzy logic is also extensively used for optimization problems in the supply chain management domain. Gunasekaran et al. [25] have deployed a triangular fuzzy quality function deployment (QFD) algorithm, Monte Carlo simulation, and a multi-objective model to optimize the total user preferences from customer responses. Their objective is to find a set of optimal solutions with respect to the performance of each supplier. Their approach intends to provide decision-making with an optimal solution in a QFD-based collaborative product design environment.

7. Summary

In this paper, we describe a fuzzy logic-based supply chain management agent which is capable of adapting to the market situation by controlling the target inventory level and profit margin. The agent is designed to take into account the level of stock holding, market trend, rate of previous successful bidding, and bidding price level of previous day. The UMTac-06 agent is designed in a way that it has both the properties of buy-to-build and build-to-order strategies. In accordance with buy-to-build strategy, the agent maintains a safe level of inventory throughout the competition. Whenever the inventory falls below the safe level, the agent immediately negotiates with the suppliers and sources the required components. The agent also assembles finished products from surplus factory cycles and components to increase the production level. In accordance with build-to-order strategy, the agent acquires components after it has secured customers' orders.

The overall results of the competitions and the performance comparison of two main strategies (heuristic and fuzzy logic) are consistent with our preliminary assumptions during the UMTac-06 agent design. In particular, they confirmed our intuition that a fuzzy logic-based strategy could deliver better results and be more resilient to different market environments. The results from our analysis demonstrate a significant improvement over the performance when the fuzzy logic-based agent is compared with a heuristic-based approach.

One of the main highlights of our study is that it shows that the fuzzy logic-based strategy for controlling inventory level not only delivers better results in factory utilization but also reduces the cost for bank interest, storage, and penalty. It also shows that the agent's strategy on bidding price setting has effectively improved the average PC selling price and success rate in bidding customer orders.

Although the UMTac-06 agent is designed to adapt to the highly dynamic market environment, it currently lacks mechanisms for projecting future prices and for predicting supply/demand in both customer and supplier negotiations. To improve this situation, we are currently investigating several alternatives to incorporate customer demand and component price prediction mechanisms into our fuzzy logic-based framework. We are also investigating the possibility of selecting customer RFQs based on multi-objectives optimization techniques.

8. Acknowledgment

The authors thank editors, anonymous referees, and Dr. Robert P. Biuk-Aghai for their insightful comments. This research was funded by the University of Macau.

References

[1] A. Petric, V. Podobnik, and G. Jezic. The CrocodileAgent: Analysis and comparison with other TAC SCM 2005 agents, In Proceedings of the AAMAS Joint International Workshop on the Trading Agent Design and Analysis and Agent Mediated Electronic Commerce (TADA/AMEC), pp. 202-205, 2006.

[2] C. Gane and T. Sarson. Structured Systems Analysis: Tools and Techniques, Prentice-Hall, 1979.

[3] C. Kiekintveld, J. Miller, P. R. Joadan, and M. P. Wellan. Controlling a Supply chain Agent Using Value-based Decomposition, In Proceedings of the Seventh ACM Conference on Electronic Commerce, pp. 208-217, 2006.

[4] D. M. Galindo, D. V. Ayala, and F. L. López. Tiancalli06: An agent for the Supply Chain Management Game 2006, In Proceedings of the International Conference on Computational Intelligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce, pp. 64-69, 2006.

[5] D. Pardoe and P. Stone. An Autonomous Agent for Supply Chain Management, in Gedas Adomavicius and Alok Gupta, editors, Handbooks in Information Systems Series: Business Computing, Elsevier, 2007, (to appear).

[6] D. Pardoe and P. Stone. Bidding for Customer Orders in TAC SCM: A Learning Approach, Lecture Notes in Computer Science, Springer, pp. 143-157, New York, 2004.

[7] D. Petrovic, R. Roy, and R. Petrovic. Supply chain modelling using fuzzy sets, International Journal of Production Economics, Elsevier, 59(1-3), pp. 443-453, March, 1999.

[8] E. Dahlgren and P.R. Wurman. PackaTAC: A Conservative Trading Agent. SIGecom Exchanges, 4(3), pp. 33-40, 2004.

[9] F. R. Lin and S. M. Lin. Enhancing the Supply Chain Performance by Integrating Simulated and Physical Agents into Organizational Information Systems, Journal of Artificial Societies and Social Simulation, 9(4), pp. 1, 2006.

[10] F. R. Lin and Y. Y. Lin. Integrating multi-agent negotiation to resolve constraints in fulfilling supply chain orders, Electronic Commerce Research and Applications, 5(4), pp. 313–322, 2006.

[11] I. Kontogounis, K. C. Chatzidimitriou, A. L. Symeonidis, and P. A. Mitkas. A Robust Agent Design for Dynamic SCM Environments, Lecture Notes in Computer Science, Springer, pp. 127-136, 2006.

[12] J. Collins, R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne and S. Janson. The Supply Chain Management Game for 2006 Trading Agent Competition. Technical Report CMU-ISRI-05-132, School of Computer Science, Carnegie Mellon University. USA, November 2005. Available from http://www.sics.se/tac/tac05scmspec_v157.pdf, accessed 16-September-2005.

[13] J. Jiao, X. You, and A. Kumar. An agent-based framework for collaborative negotiation in the global manufacturing supply chain network. Robotics and Computer-Integrated Manufacturing, 22(3), pp. 239-255, 2006.

[14] J. M. Swaminathan, S. F. Smith, and N. M. Sadeh. Modeling Supply Chain Dynamics: A Multiagent Approach, Decision Sciences, 29(3), pp. 607-632, July 1998.

[15] J. Wang and Y.-F. Shu. Fuzzy decision modelling for supply chain management, fuzzy sets and systems, Elsevier, 150(1), pp. 107-127, 2005.

[16] K. Decker and J. Li. Coordinated hospital patient scheduling, In Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98), pp. 104–111, 1998.

[17] L. Penserini, M. Kolp, and L. Spalazzi, Social-oriented engineering of intelligent software, WIAS: Web Intelligence and Agent Systems: An International Journal, IOS Press, (5)(2007), pp. 69–87, 2007.

[18] L. Zadeh. Outline of a new approach to the analysis of complex system and decision process, IEEE Transactions on Systems, Man, and Cybernetics, SMC, 3(1), pp. 28-44, 1973.

[19] M. Benish, A Greenwald, I Grypari, R. Lederman, V. Naroditskiy, and M. Tschanta. Botticelli: A Supply Chain Management Agent Designed to Optimize under Uncertainty. SIGecom Exchange 4(3), pp. 29-37, 2004.

[20] M. He and N. R. Jennings. SouthamptonTAC: An adaptive autonomous trading agent, ACM Transactions on Internet Technology, 3(3), pp. 218-235, August 2003.

[21] M. He, A. Rogers, E. David, and N. R. Jennings. Designing and Evaluating an Adaptive Trading Agent for Supply Chain Management Applications. IJCAI-05 Workshop on Trading Agent Design and Analysis, Lecture Notes in Computer Science, Springer, pp. 140-156, 2005.

[22] M. P. Wellman, A. Greenwald, P. Stone, and P. R. Wurman. The 2001 trading agent competition. In Proceedings of the 14th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI-02), pp. 935-941, Edmonton, Alberta, Canada, 2002.

[23] M. Wang, J. Liu, H. Wang, W. K. Cheung, and X. Xie. On-demand e-supply chain integration: A multi-agent constraint-based approach, Expert Systems with Applications, 34(4), pp. 2683-2692, 2008.

[24] M. Wooldridge and N. Jennings. Intelligent agents: theory and practice. Knowledge Engineering Review, 10(2), pp. 115-152, 1995.

[25] N. Gunasekaran, S. Rathesh, S. Arunachalam, and S.C.L. Koh. Optimizing supply chain management using fuzzy approach, Journal of Manufacturing Technology Management, 17(6), pp. 737-749, 2006

[26] N. Julka, I. Karimi, and R. Srinivasan. Agent-based supply chain management – 2: a refinery application, Computers and Chemical Engineering, 26(2002), pp. 1771-1781, Elsevier, 2002.

[27] N. Julka, R. Srinivasan, and I. Karimi. Agent-based supply chain management – 1: framework, Computers and Chemical Engineering 26(2002), pp. 1755-1769, Elsevier, 2002.

[28] P. W. Keller, F. Duguay, and D. Precup. RedAgent: Winner of TAC SCM 2003, SIGecom Exchanges, 4(3), pp. 1-8, 2004.

[29] P. W. Keller, F.-O. Duguay, and D. Precup. RedAgent: Winner of TAC SCM 2003. SIGecomExchanges, 4(3):1-8, Winter 2004.

[30] R. Dong, T. Tai, W. Yeung, and D. C. Parkes. HarTAC – The Harvard TAC SCM'03 Agent. In Proceeding of the

AAMS-04 Workshop on Trading Agent Design and Analysis, pp. 1-8, New York, 2004.
[31] R. Zimmermann, S.Winkler, and F. Bodendorf. Agent-Based Supply Chain Event Management - Concept and Assessment, In Proceedings of the 39th Annual Hawaii International Conference on System Sciences, vol.2, pp. 40a, 2006.
[32] S. Buffett. A Markov Model for Inventory Level Optimization in Supply-Chain Management, In Proceedings of the Eighteenth Canadian Conference on Artificial Intelligence (AI2005), pp. 133-144, 2005.
[33] S. K. Yung, C. C. Yang, A. S. M. Lau, and J. Yen. Applying multi-agent technology to supply chain management, Journal of Electronic Commerce Research, 1(4), pp.119-132, 2000.
[34] S. Sun, V. Avasarala, T. Mullen, and J. Yen. PSUTAC: a trading agent designed from heuristics to knowledge. In Proceedings of the AAMAS-04 Workshop on Trading Agent Design and Analysis, pp. 15-20, New York, 2004.
[35] S.-F. Lou and Y.-W. Si. Fuzzy adaptive agent for supply chain management, In Proceedings of the 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-06), Hong Kong, China, pp. 313-320, IEEE Press, 2006.
[36] T. J. Strader, F. R. Lin, and M. J. Shaw. Simulation of order fulfillment in divergent assembly supply chains, Journal of Artificial Societies and Social Simulation, 1(2), pp. 4, 1998.
[37] T. Wagner, V. Guralnik, and J. Phelps. TAEMS agents: enabling dynamic distributed supply chain management, Electronic Commerce Research and Applications, 2(2), pp.114-132, 2003.
[38] X. Bo and W. Zhiming. Modeling of supply chain: a multi-agent approach, in Proceedings of the American Control Conference, pp. 1056-1061, 2003.
[39] Y. Fu, R. Piplani, R. D. Souza, and J. Wu. Agent-based simulation applications: multi-agent enabled modeling and simulation towards collaborative inventory management in supply chains, in Proceedings of Winter Simulation Conference 2000, pp. 1763-1771, 2000.
[40] Y.-W. Si, D. Edmond, M. Dumas, and C. U. Chong. Strategies in Supply Chain Management for the Trading Agent Competition. Electronic Commerce Research and Application, 6(4): pp. 369-382, 2007.