Strategies for a Sales Component of an Intelligent Agent for the Supply Chain Management Game of the 2003 Trading Agent Competition

Elena V. Kryzhnyaya

yelena@cs.umn.edu

University of Minnesota

August 18, 2003

Abstract

This paper considers two different strategies for a sales component of an intelligent agent designed for the supply chain management (SCM) game (www.sics.se/tac), which is a new addition to the 2003 Trading Agent Competition (TAC). TAC-SCM game involves six software agents attempting to maximize profits by manufacturing computers in a simulated market economy. TAC-SCM intelligent agents participate simultaneously in two simulated markets: a market of raw materials (RM) and a consumer market. This paper introduces two strategies for competing in the consumer market, without considering issues such as acquiring components and assembling products. The primarily goal of the sales strategies we introduce is to maximize revenue by selling out the existing finished goods (FG) inventory built by other components of the intelligent agent. The automation of the sales strategy is an interesting problem because it is one of the most important factors affecting the overall performance of an agent participating in an electronic market. Successful automation of the sales strategy requires a synthesis of several fields, including economics, game theory, artificial intelligence, machine learning, multiagent systems, statistics, and probability.

The primary contribution of this paper is its proposed approaches to the automation of the sales strategy and its extensive experimental analysis and comparison of the strategies in the TAC-SCM simulation environment. We describe our initial implementation of the sales algorithms for a TAC-SCM agent, and present experimental results that seem to indicate that this implementation is at least as sophisticated as those of other teams. These experiments also illustrate areas where our agent's performance could be improved. We conclude with a

discussion of future research directions.

Acknowledgments

I would like to express my gratitude to all the people who supported and encouraged me during the project. Thank you to my advisor Maria Gini for her guidance, motivation and invaluable suggestions at all stages of this project and the course of my graduate studies experience. Thank you to all members of TAC-SCM research team including Maria Gini, John Collins, Stephen Damer, Colin McMillen, Alexander Babanov, and Amrudin Agovic for their had work, timely help, valuable advises and support. Thanks to my committee Maria Gini, John Collins and Stergios Roumeliotis for your assistance and advice.

Thank you to my sister and all of my friends for making my life more colorful and fulfilling. Special thanks to my family, especially my parents. I would never be the person I'm today without their blessings, continued support, push, inspiration, patience, and encouragement through all periods of my life.

Contents

Chapter 1

Introduction

Recently, the organization of competitive simulation environments has been increasingly utilized as a publicly shared testbed for the development of multiagent systems and autonomous intelligent agents. Perhaps the most well-known multiagent competition is the Robot World Cup Initiative (RoboCup) , a robotic soccer competition that specifically emphasizes autonomous multiagent collaboration [2]. Another competition sponsored by the RoboCup Federation is RoboCup Rescue [3], in which teams of autonomous or teleoperated robots attempt to find victims in an urban search and rescue course. Researchers interested in autonomous bidding agents started the Trading Agent Competition (TAC) in 2000 [6]. The original TAC game involves agents competing to satisfy clients' travel and entertainment desires. Agents are responsible for buying airline tickets, reserving hotels, and finding entertainment packages suitable to each client's desires. The Trading Agent Competition has been an annual event since 2000. [4]

Stone [5] discusses multiagent competitions, including RoboCup and TAC. He gives an overview of the rules of these competitions, and notes some similarities and differences between the two. Most importantly, Stone shares his views on the benefits and drawbacks of such competitions. These views are based on his participation in numerous instances of the RoboCup and TAC competitions. He claims that there are many ways in which scientific

progress can be hindered by an organized competition, including: obsession with winning, domain-specific solutions, barriers to entry, restrictive rules, and invalid evaluation conclusions. On the other hand, there are also many ways in which scientific progress can be advanced by an organized competition, by providing research inspiration, providing deadlines for creating complete working systems, providing a common platform for testing and exchanging ideas, encouraging continual improvement of solutions, and encouraging flexible software and hardware. Stone concludes that the benefits of RoboCup and TAC outweigh the hazards, but emphasizes that the competition results alone are not scientifically conclusive. He recommends that implementations of novel approaches to the problems posed by RoboCup and TAC be tested not only in competition, but also in controlled, empirical experiments. [4]

A new game for the 2003 Trading Agent Competition (TAC-03) has been proposed by researchers at the Carnegie Mellon University e-Supply Chain Management Laboratory and the Swedish Institute of Computer Science (SICS) [1]. This game involves a supply chain management scenario in which agents attempt to maximize profits by manufacturing personal computers (PCs) to sell to customers. This new supply-chain management game is called TAC-SCM; the original game is now referred to as TAC Classic. Supply chain management is concerned with planning and coordinating the activities of organizations across the supply chain, from raw material procurement to finished goods delivery. In today's global economy, effective supply chain management is vital to the competitiveness of manufacturing enterprises as it directly impacts their ability to meet changing market demands in a timely and cost effective manner. With annual worldwide supply chain transactions in the trillions of dollars, the potential impact of performance improvements is tremendous. While today's supply chains are essentially static, relying on long-term relationships among key trading partners, more flexible and dynamic practices offer the prospect of better matches between suppliers and customers as market conditions change. Adoption of such practices has however proven elusive, due to the complexity of many supply chain relationships and

the difficulty in effectively supporting more dynamic trading practices. TAC-SCM 2003 was designed to capture many of the challenges involved in supporting dynamic supply chain practices. [1]

There are a number of reasons why we believe the TAC-SCM competition is interesting. Agents participating in a TAC game must base their decisions on limited information about the state of the market and the strategies of other agents. Agents must simultaneously compete in two separate but interrelated markets: the market from which the agents must buy their supplies and the market to which the agents must sell their finished products. Agents have a large number of decisions to make in a limited time, so the computational efficiency of the decision-making process is paramount. We believe that effective solutions to the TAC SCM game will be multidisciplinary, as there is substantial related work in the fields of industrial operations research, economics, game theory, artificial intelligence, multiagent systems, computational complexity theory, statistics, and probability. [4]

Chapter 2 describes different approaches to TAC-SCM intelligent agent design that lead to different formulations of the sales manager's goals. Two proposed approaches to sales strategy design, called MaxEProfit and DemandDriven strategies, are described in chapters with corresponding names. In Chapter 5 the results of an experimental analysis to compare our proposed strategies in the TAC-SCM simulation environment are presented. We conclude by discussing directions in which future work is likely to progress.

Chapter 2

Game Overview and Agent Organization

The TAC-SCM game simulates an environment where six software agents compete for customer orders and for procurement of a variety of raw material components over a period of several simulation days. Each day customers issue requests for quotes (RFQ) and select what to buy from quotes submitted by raw material (RM) supplier agents, based on delivery dates and prices. The agents are limited by the capacity of their assembly lines and have to procure raw material components from a set of eight suppliers. Each day a trading agent has to make decisions about which raw material to purchase, which finished goods (FG) to assemble, and which customer RFQs to respond to and what price to offer. The agent designed by the team from the University of Minnesota, MinneTAC agent, is built using a component-based architecture based on Avalon. Each component has different responsibilities and goals. The Avalon framework ensures communication and collaboration between different components through event postings and method invocations. Responsibilities to make important daily decisions are split among different components, i.e. the RM Supplier Manager decides which raw material to purchase, the Assembly Manager - which finished products to assemble, and the Sales Manager - which customer RFQs to respond to and

what price to offer.

There could be several approaches to implement a TAC-SCM agent, that lead to different formulations of the Sales Manager's goals and strategies. The first approach could be called customer-demand-driven or build-to-order, the second - supply-driven.

The customer-demand-driven approach assumes that the bottleneck is in the customer demand. More formally a bottleneck is a factor that limits the number of PCs produced in a TAC-SCM game. A bottleneck on the customer side means that all aspects of an agent's processing should depend on the customer orders awarded. This assumption determines both the preferable order of the agent's actions and the strategies of each component including the SalesManager. The SalesManager's goal in this approach is to maximize its profit based on estimated or nominal product costs. Once a profitable order has been awarded by a customer, the goal of the other components is to acquire the necessary raw materials and build the requested products before the order is due. DummyAgent, the agent provided by the game organizars as part of the tac-agentware/ framework, uses this approach. The advantage of this approach is the agent's flexibility to stop doing business in an unprofitable market environment with minimal losses due to unused raw material inventories. However, this approach has several disadvantages as well. One of the drawbacks of customer-demand-driven strategy is inability to respond to short lead-time customer requests. Also in the current TAC-SCM'03 game setup, agents could receive a significant (up to 50%) discount for RM components ordered in advance. This fact causes the major disadvantage of a build-to-order approach that is higher product costs resulting in the inability of such an agent to successfully compete for customer orders with agents that use a different strategy. This fact, along with the fact that many games have a RM-supplies bottleneck, is the reason why the vast majority of teams participating in TAC-SCM'03, including MinneTAC, have chosen the second approach, which is supply-driven.

The second approach could be categorized as supply-driven or bid-from-current-inventory. This approach involves pre-estimating the average overall use of RM materials during a game

setup with average customer demand, and pre-ordering the major part of the needed RM inventories at the beginning of the game. The agent has to make this decision blindly on the first day of the game since by the time it has any information about the initial demand it usually is already too late to receive a good discount for the RM supplies. Hence, this decision should be pre-configured, based on previously observed games. By ordering on the first day of the game, agent could decrease its product costs by up to 50% compared to the nominal price when the parts are ordered from component suppliers at full price. Once an agent has acquired all the RM components, it is bound by the goal to sell all the products it could produce before the predetermined end of the game, even if it means selling at a big loss. This is a major disadvantage of the supply-driven approach. This downside was mainly caused by the game setup itself allowing to get a huge discount on the RM components only by ordering during the first day of the game. So, every team had to make a tradeoff decision about what startegy to use. Choosing the supply-driven approach resulted in higher business profitability in games with the bottleneck on the RM suppliers' side and higher business losses in games with the bottleneck on the customer side. Since a lot of the games observed during the qualifying and seeding rounds had a bottleneck on the supplier side, the MinneTAC team decided to go with supply-driven approach. This decision resulted in a sales manager goal to sell out existing FG inventory before the end of the game, even if it meant selling at a loss, maximizing the overall profit as much as possible.

Based on this goal two sales manager strategies, MaxEProfit and DemandDriven, were implemented and compared. Both strategies were designed assuming that the probability of a customer order could be influenced by the following parameters: reserve price, demand, quantity, lead-time, penalty, product type and price specified by the customer agent. So, the probability of order is a function of several variables.

It is obvious that some parameters could influence the probability of order more than the others. To determine the influence of each of the parameters on the probability of order curve, the data from several TAC-SCM games were analyzed.

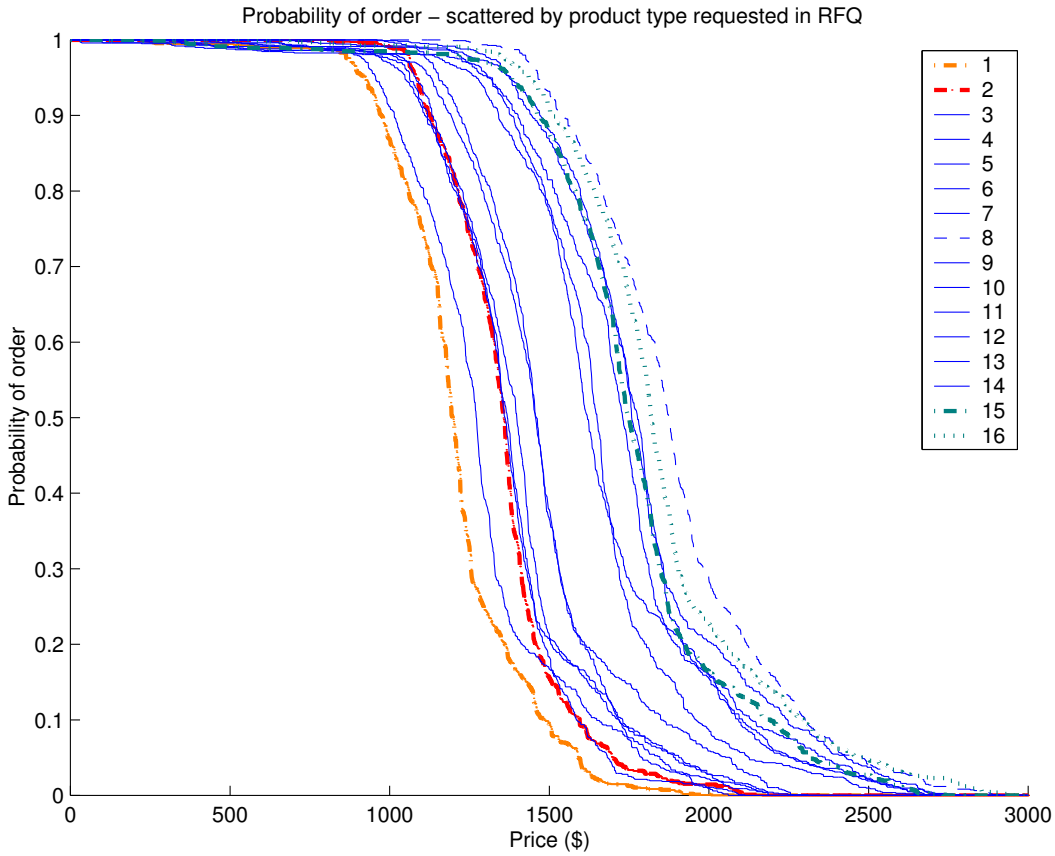

Figure 2.1: Probability of order for different product types requested in a RFQ.

The analysis uncovered that parameters such as quantity, lead-time, and penalty have an insignificant influence on the probability of order curve. Since these results could vary a lot depending on other agents' strategies, they couldn't be considered as holding for future TAC-SCM game settings. Figure 2.1 shows that the influence of the product type parameter on the probability of order curve could be considered significant. The insignificant influence of the quantity parameter, for instance, is easy to notice by compering Figure 2.1 with Figure 2.2.

Figure 2.3 shows the probability of an order as a function of the offered price for two different customer demand levels: a low demand situation when the average product demand was 2037.75 PCs per day, and a high demand situation when the average product demand was almost twice as high as during a low demand period - 3956.77 PCs per day. Figures 2.3,

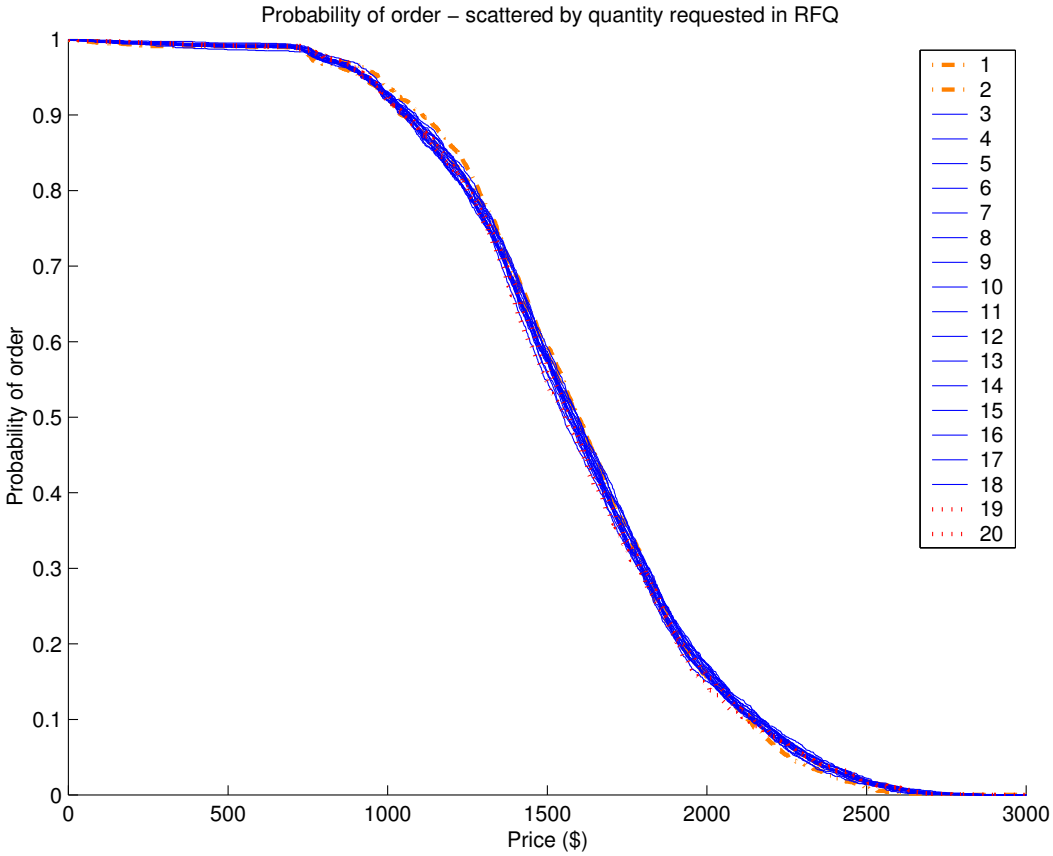

Figure 2.2: Probability of order for different product quantities requested in a RFQ.

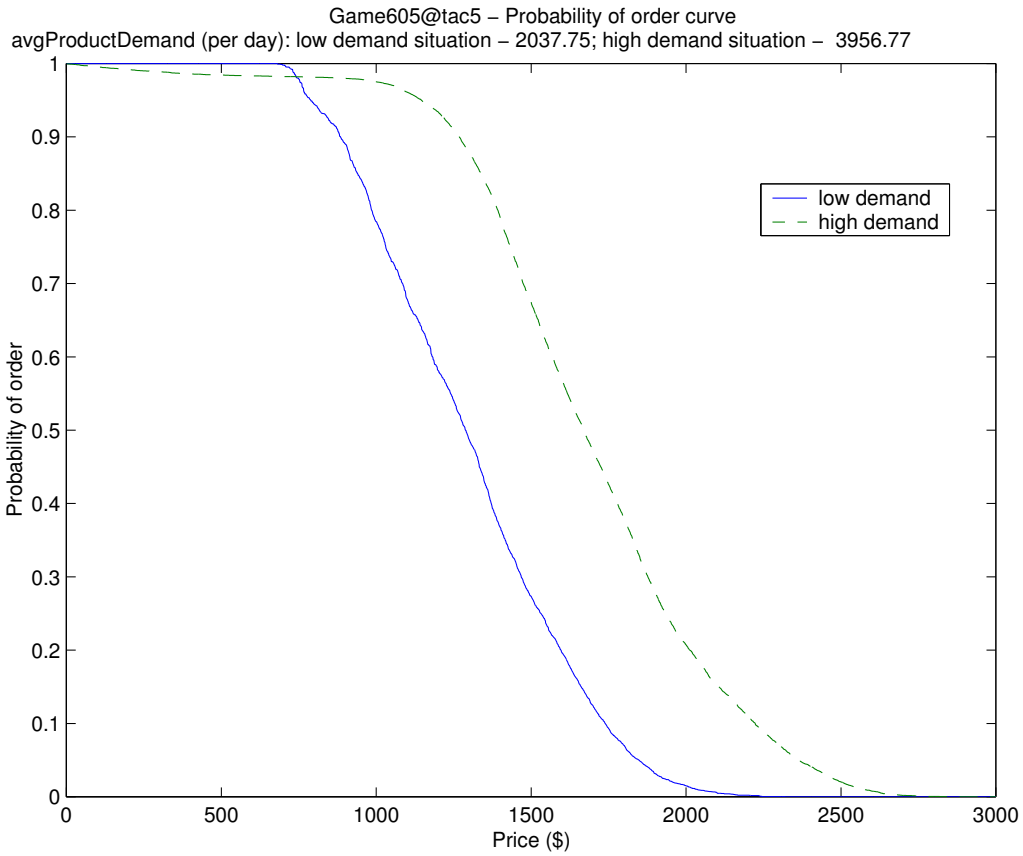

Figure 2.3: Probability of an order for different customer demand levels.

2.1, and 2.2 were generated from the TAC-SCM seeding round's game 605@tac5.sics.se full data analysis (see Figures A.1 and A.2 for game results).

Chapter 3

MaxEProfit Sales Strategy

As it was mentioned before, the Sales Manager component's responsibilities are to decide which customer RFQs to respond to and which price to offer. Let us start by explaining how the MaxEProfit Sales strategy decides which price to offer.

Any acceptable sales strategy must be able to accurately estimate the probabilities that customers will accept the offers. As mentioned in previous chapter, there is a variety of factors that may influence this probability, including the offer price, lead-time, reserve price, penalty, quantity, product type specified in the RFQ, current customer demand level, availability of supplies, and the strategies of other agents. These factors are partially dependent on one another and not necessarily known to the agent's developers *a priori*. Even though the probability of order as a function of several variables could be initially determined based on previous games' data analysis, this doesn't eliminate the need for on-line machine learning. The ability of the sales strategy to learn during the game could play a crucial role since contingencies such as other agents' strategies could significantly affect the probability of order function. The MaxEProfit Sales strategy assumes that the probability of order depends only on the offer price and on such values specified in the RFQ as product type, quantity, lead time, reserve price, and penalty. It doesn't take into account other external factors that could affect the probability of order and it doesn't model other agents' pricing behavior.

MaxEProfit Sales Manager stores a 6-dimensional *OrderProbability* matrix:

$OrderProbability = offer\_price \times quantity \times lead\_time \times reserve\_price \times penalty \times producttype.$

Each entry in *OrderProbability* contains the probability that customer will accept an offer with given offer price. To keep *OrderProbability* small the values of offer prices, reserve prices, and penalties were partitioned into ranges. Initially MaxEProfit Sales Manager assumes that customer will accept offers with any offer price equal or lower than the reserve price specified in the RFQ and all those values in *OrderProbability* are set to 1.0. Each day MaxEProfit Sales Manager updates the appropriate elements of *OrderProbability* with the configurable *learningRate* parameter, based on whether its offer for a particular customer RFQ was accepted (true=1) or not (false=0).

$$OrderProbability(element) = (1 - learningRate) \times OrderProbability(element) + \\ + learningRate \times order\_acceptance\_fact. \quad (3.1)$$

Such on-line update of the *OrderProbability* matrix based on every independent event may not be justified from the probability theory viewpoint. Another approach might be to use a 6-dimensional *OrderProbability* matrix that stores two values in each element: the number of offers the agent has made on RFQs with the associated parameter values, and the number of times that these offers have been accepted by customers. The appropriate slots in this matrix can then be updated every time a bundle of offers is sent or a bundle of orders is received. To calculate the probability of acceptance we'll just need to divide the number of orders by the number of offers [4]. However, the biggest problem with this approach is that this method considers all offers equally. Since the state of the market changes over the course of the game, the results of the most recent offers are much more relevant.

So, MaxEProfit Sales Manager learns the mapping from offer prices to the probability that a customer will accept the offer. Given this information MaxEProfit Sales Manager

suggests a price that will maximize the expected profit from the order:

$$E[profit(x)] = profit \times P(acceptance(x)). \tag{3.2}$$

in constraint that $price \geq targetAveragePrice(type\_of\_product)$ if the reserve price specified in the RFQ permits it. Since interest rates are an insignificant factor in the current TAC-SCM'03 game setup, they are not taken into account at this point. It's also worth noting that the current MaxEProfit sales strategy calculates profit in equation 3.2 assuming nominal product prices. $targetAveragePrice$ is the average price that the agent realistically could and is trying to achieve in the current market situation. This constraint is necessary to ensure that other agents are not undercutting MinneTAC on a profitable RFQs because it is bidding too high, and at the same time doesn't allow the agent to lower prices more than it is necessary in a current market situation. The $targetAveragePrice$ parameter helps to find an upper price limit satisfactory for undercutting other agents on profitable RFQs and ensures that MinneTAC is not left satisfied with winning only unprofitable RFQs. So, every day when the agent has received a bundle of customer RFQs, it constructs offers as follows:

1. Determines a price for every RFQ based on the algorithm described above.

2. Sorts all potential offers in descending order by the profit margin that the agent will obtain in case of an order. $profit\_margin = \frac{(price - cost)}{price}$

3. Sends an offer if an agent has the requested quantity of PCs in FG inventory but reserve for order only $reserve\_quantity(x) = requested\_quantity(x) \times P(actual\_acceptance(x))$, where $P(actual\_acceptance(x))$ is the actual offer acceptance rate, updated each day for each type of product.

4. Repeats the previous step until all current uncommitted FG inventory has been offered for sale or offers for all customer RFQs have been generated.

The periodic market reports produced by the server are used by the MaxEProfit Sales Manager to adjust its strategy. Every 20 simulation days the agent receives a market report and adjusts $targetAveragePrice$ to be 8% higher or lower than $MarketAverage$ depending on the time left in the game and on the levels of uncommitted FG inventories. The MaxEProfit Sales Manager tries to maximize overall profit without worrying about grows of FG inventory until it realizes that it needs to start selling off the inventory before the end of the game. To determine a *sell-off point* the agent keeps track of the moving average number of products build each day as well as moving average demand level for each product type. The MaxEProfit Sales Manager determines a *sell-off point* for a particular type of product when the following inequality becomes valid:

$$\begin{aligned} sell-offpoint : available\_FG\_inventory(product\_type) + \\ +products\_produced(product\_type) \times number\_of\_days\_left \geq \\ \geq demand(product\_type) \times number\_of\_days\_left. \end{aligned} \tag{3.3}$$

Only when MaxEProfit Sales Manager detects a *sell-off point* for a product, the point when existing FG inventories should be sold out at any positive price, it keeps gradually lowering $targetAveragePrice$ till it hits an offer acceptance rate close to 100%. Theoretically this approach should work reasonably well even in a customer-demand-limited games but in practice, other agents faced with the same problem are willing to lower their prices dramatically too. So, usually towards the end of a customer-demand-limited game, the average market prices drop very fast. As a result PCs are sold for unreasonably low prices toward the end of such a game. This lead us to the experimental conclusion that in a customer-demand-limited games it is more profitable to lower prices reacting to the current market situation as soon as possible without allowing any increase of the FG inventories. However, in supply-limited games this heuristic causes an agent to lower prices too early. In games when demand level swings a lot during the game, it also pays off to accumulate FG inventories and wait for

better market conditions to sell them out.

In an attempt to to find a more flexible heuristic the second, DemandDriven Sales Manager strategy was formulated, implemented, and tested.

Chapter 4

DemandDriven Sales Strategy

The DemandDriven Sales strategy is different from MaxEProfit in several aspects. First of all, it uses a different approach to learn the mapping from offer prices to the probability that a customer will accept the offer. Even though MaxEProfit doesn't take into account all factors that could affect the probability of an order, the size of the *OrderProbability* matrix was congruent or sometimes exceeded the total number of customer RFQs send during one game. This fact lead to uselessness of *OrderProbability* learning during some games. To increase its usefulness, the *OrderProbability* matrix was modified in the DemandDriven Sales strategy approach. As a detailed analysis of the seeding rounds' games uncovered, such factors specified in the customer RFQ as quantity, lead-time, and penalty have an insignificant influence on the probability of order curve (see Chapter 2 for more details). Again, it is worth mentioning that these results could vary a lot depending on the other agents' strategies, hence they couldn't be considered as holding for future TAC SCM game settings. However, with currently participating agents we decided to assume that quantity, lead-time, and penalty are not influencing the probability of order curve. This assumption even further reduced the size of *OrderProbability* matrix and increased its usefulness. The DemandDriven Sales Manager stores a 5-dimensional *OrderProbability* matrix $OrderProbability = offer\_price \times customer\_demand \times lead\_time \times reserve\_price \times product\_type$. Each entry in

OrderProbability contains the probability that the customer will accept an offer with a given offer price. To keep *OrderProbability* small the values of offer prices, reserve prices, lead-time, and customer demand level were partitioned into ranges. The lead-time parameter was subdivided into two ranges: short and long lead-time customer RFQs. Customer demand level was subdivided into three ranges: low, medium, and high, based on analysis of previous game data. Initially *OrderProbability* is pre-populated with the probability of order values generated based on the analysis of several previous seeding rounds' games. DemandDriven Sales strategy assumes that the initial *OrderProbability* values should be reasonably accurate and hence it doesn't use on-line learning to update these values. For better fine-tuning during the game the DemandDriven strategy assumes that the initial *OrderProbability* curve could shift during the game without changing its shape. *Multiplier* is a factor allowing the DemandDriven strategy to properly adjust the shift of the probability of order curve during the game if needed. Multiplier is initially set to 1.0 assuming that no adjustment is needed.

The other difference of the DemandDriven Sales strategy from MaxEProfit is that it tries to ensure that the FG inventory is sold out towards the end of the game from the very first day. Based on the information about the product demand, the available FG inventory, and products build each day, it calculates the *targetAcceptanceRate* for each type of product:

$$targetAcceptanceRate = \frac{available\_FG\_inventory(product\_type) + products\_produced(product\_type) \times number\_of\_days\_left}{OptimisticDemandEstimate} \quad (4.1)$$

During the first 170 days of the 220 day long game, the DemandDriven strategy is optimistic about the demand level and takes into account the chance that it could go up later in the game. So, during first 170 days of the game:

$$OptimisticDemandEstimate = current\_demand(product\_type) \times number\_of\_days\_left + (number\_of\_days\_left - 50) \times current\_demand(product\_type) \quad (4.2)$$

During the last 50 days of the game, the DemandDriven Sales strategy calculates $targetAcceptanceRate$ based on the current demand without being optimistic about it. After calculating $targetAcceptanceRate$, the DemandDriven Sales Manager suggests prices that it believes should make the actual acceptance rate close to $targetAcceptanceRate$ by using a pre-specified dependency curve between the probability of an order being accepted, given the demand level, reserve price, lead time, and product type.

DemandDriven goes through exactly the same steps as MaxEProfit to generate offers. Every day when the agent has received a bundle of customer RFQs, it generates offers as follows:

1. Determines a price for every RFQ based on the algorithm described above.

2. Sorts all potential offers in descending order by profit margin that the agent will obtain in case of an order. $profit\_margin = \frac{(price - cost)}{price}$

3. Sends an offer if an agent has requested a quantity of PCs that in the FG inventory but reserve for the order only $reserve\_quantity(x) = requested\_quantity(x) \times P(actual\_acceptance(x))$, where $P(actual\_acceptance(x))$ is the actual offer acceptance rate, updated each day for each type of product.

4. Repeats the previous step until all current uncommitted FG inventory has been offered for sale or offers for all customer RFQs has been generated.

The other difference of DemandDriven from MaxEProfit is that after each day it calculates the $actualAcceptanceRate$ and compares it with the $targetAcceptanceRate$. If $actualAcceptanceRate$ is higher than the $targetAcceptanceRate$, it means that the prices were lower than the agent wanted. In this case DemandDriven increases the $Multiplier$ which corresponds to a shift of the probability of order dependency curve to the right. If $actualAcceptanceRate$ is lower than the $targetAcceptanceRate$ that means that the prices were higher than the agent wanted. In this case, DemandDriven decreases the $Multiplier$

which corresponds to a shift of the probability of order dependency curve to the left. Hence, *Multiplier* is used for more fine-tuning during the game since the probability of order curve in every game could be different from the one constructed based on the data from previous games.

Chapter 5

Experimental Analysis and Comparison

To compare the two Sales Manager strategies described above a number of games were analyzed. As our experimental results uncovered, both sales strategies have advantages and disadvantages. The opportunities to explore different strategies that might have the advantages of both without having their disadvantages are left as open questions for future research. This section provides a detailed comparison of the two currently implemented Sales Manager strategies, MaxEProfit and DemandDriven.

As our experiments uncovered, the DemandDriven Sales strategy works better in situations when customer demand is lower than overall market supply, i.e. when customer demand is 100% satisfied and participating agents are forced to keep lowering their prices to win customer orders. Examples are the games (agent rudy run DemandDriven Sales strategy, rudy0 - MaxEProfit Sales strategy): 1181@tac5.sics.se, 1336@tac6.sics.se, and 1331@tac6.sics.se (see Figures A.5 and A.6 for game results). The DemandDriven Sales Manager reacts to changed market conditions and lowers prices faster then the MaxEProfit Sales strategy. So, the DemandDriven Sales strategy could be thought as being not as optimistic about future customer demand as the MaxEProfit Sales strategy. On the other hand, the MaxEProfit

Sales strategy works better (ends up with higher average prices) when customer demand is permanently or temporarily higher than all participating agents could supply (i.e. when customer demand is permanently or temporarily not 100% satisfied). So, the DemandDriven Sales strategy is good at lowering prices faster (hence it looses in profit in high demand games), whereas MaxEProfit Sales strategy refrains from reacting to market conditions right away and waits to see if customer demand will exceed the overall market supply again, so that it could sell the accumulated FG inventory at better prices. If customer demand exceeds the supply later in the game, MaxEProfit ends up with higher overall revenue than that of DemandDriven and the wait turns out to be justified. However, if demand doesn't exceed the supply during the rest of the game, MaxEProfit strategy usually ends up with larger unsold FG inventories and lower overall revenue than that of DemandDriven and the wait turns out to be completely unjustified. So, since MaxEProfit tries to sell-off accumulated FG inventory only towards the end of the game, in games where supplies level exceeds customer demand it usually ends up with lower overall profit and larger unsold FG inventories because it has waited for a better market conditions and as a result started to compete for customer orders when average market prices were lower. Examples are the games (agent rudy0 run MaxEProfit Sales strategy, rudy - DemandDriven): 1180@tac5.sics.se, 1173@tac5.sics.se (see Figures A.3 and A.4 for game results). So, the MaxEProfit Sales strategy constantly beats the DemandDriven Sales strategy in games where customer demand exceeds the supply (where everybody finishes positive) and loses to it in games where supply exceeds the demand (where everybody finishes negative).

Figure 5.1 and Figure 5.2 on the example of the game 1180@tac5.sics.se illustrate that DemandDriven starts to lower its prices adapting to the market environment faster then MaxEProfit. In this game the MaxEProfit Sales strategy's delay to compete for customer orders turned out to be justified. As we can see from Figure 5.2, after day 130 customer demand became higher that overall market supply, which led to significant increase in the average market prices. As could be seen from Figure 5.1, MaxEProfit was selling during

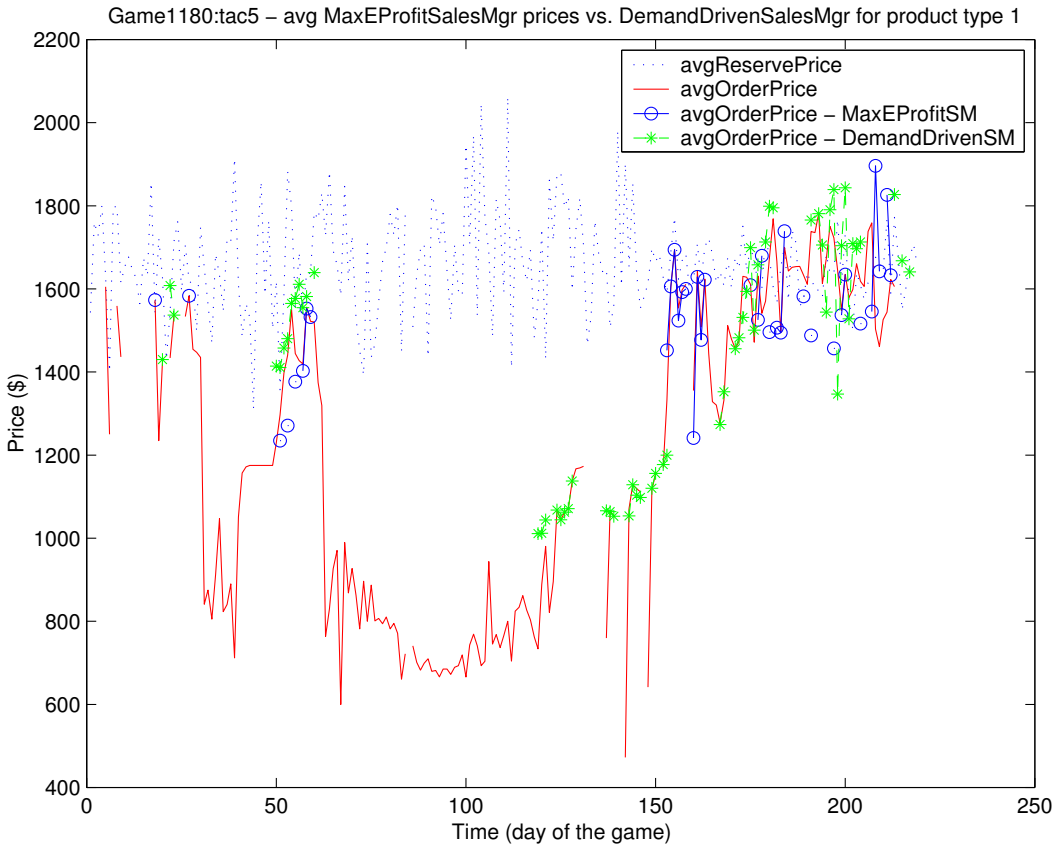

Figure 5.1: avg DemandDrivenSalesMgr vs. MaxEProfitSalesMgr prices for product type 1.

more profitable game periods and consequently achieved a higher overall renenue at the end of the game.

The real difference between these two sales strategies depends on the extent to which customer demand exceeds the supply. The last factor is hard to predict since it differs a lot from game to game depending on how other participating agents deal with RM Suppliers. Once demand becomes fully satisfied (100%) the overall market prices drop dramatically. Both strategies recognize this situation but DemandDriven Sales Manager reacts faster, which sometimes is too fast because other participating agents could run out of FG inventories, customer demand will exceed the overall supply again and hence the average market prices will go up again.

After comparing these two sales strategies, some obvious questions come to mind. What is the optimal time to wait for the demand to exceed the supply again? How to determine this

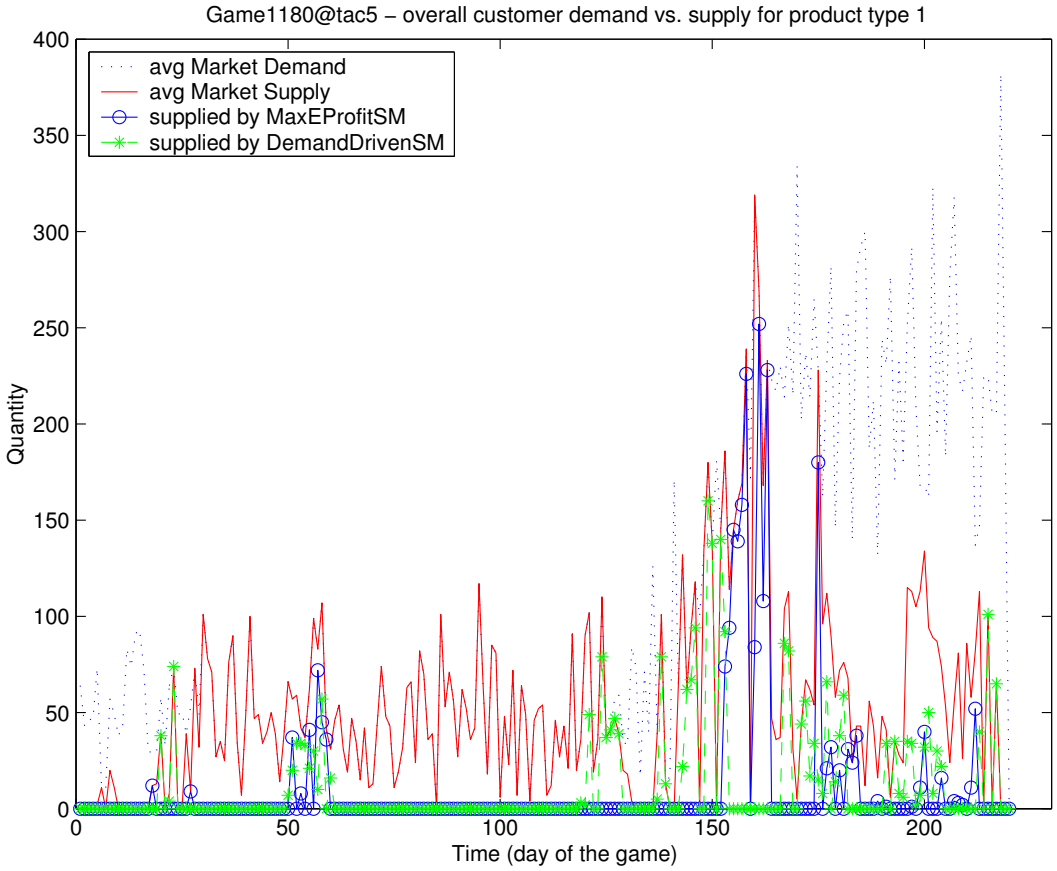

Figure 5.2: Overall customer demand vs. supply for product type 1.

time? How to determine the optimal time to start competing for customer orders once supply has exceeded customer demand? Does an agent have any direct or indirect information to estimate production capabilities of the other participating agents? These questions are left open for future research of the topic.

Chapter 6

Conclusions and Future Work

As discussed earlier, the customer-demand-driven approach to agent organization is designed and works well in games with pure customer demand bottleneck whereas the supply-driven approach is designed and works well in games with pure raw materials supply bottleneck. However, as our experiments showed, most of the time the TAC-SCM games have no unique bottleneck, which forces the participating agents to deal with decision contingences and trade-off decisions under uncertainty. Our current implementation was designed assuming that the number of supply-limited games is greater than that of customer-demand-limited games. This assumption was valid during most of the games we have observed before the semi-finals TAC-SCM'03 rounds. However, games during the semi-finals and finals were mostly customer-demand-limited, which lowered the MinneTAC score dramatically. To improve the situation in future TAC-SCM games the organization of the MinneTAC agent, the chosen precedence of actions as well as the strategy of each component should be revised to perform better in customer-demand-limited games. Apparently, introducing strategic behavior that performs better in customer-demand-limited games would worsen the performance of MinneTAC agent in supply-limited games. Hence, the goal of future research is to find a reasonable trade-off organization and strategies that would result in acceptable agent performance in both types of games. Moreover, since the market limiting factor sometimes

changes during a single game, the goal of future research may also be to increase the autonomy of MinneTAC agent by designing and implementing mechanisms to recognize current and predict future bottlenecks of a game and adapt to changing market situations more flexibly.

Taking this into account, let us discuss several ideas for possible strategic improvements in future versions of MinneTAC agent.

By analyzing the semi-finals games data, it could be concluded that MinneTAC Demand-Driven sales strategy performed reasonably well. There were two games, 1398@tac6.sics.se (see Figures A.7 and A.8 for game results) and 1402@tac6.sics.se when MinneTAC's optimistic expectation that the demand will increase in future wasn't justified. Due to being optimistic MinneTAC started to lower its prices trying to sell off accumulated inventories later in the game then other more pessimistic (and hence more realistic in this case) agents did. Since the average market prices were gradually decreasing during the game, waiting to sell off the accumulated inventories caused MinneTAC to sell at worse overall prices than it could have sold if started to lower prices faster. However, this waiting is often justified since customer demand could increase later in the game or other agents could run out of uncommitted FG inventories causing the average market prices to increase. If this happens, MinneTAC will be selling at better overall prices than other agents do. Obvious examples of such games could be semi-final games 1401@tac6.sics.se (see Figures A.9 and A.10 for game results), 1403@tac6.sics.se, and 1405@tac6.sics.se.

These experimental results raised other questions. What is the best way to estimate future customer demand during a game? In my opinion, one of the things that could be improved in future versions of both Sales Manager's strategies is a mechanism for predicting future demand as well as a tactic of being optimistic about it. As the formula 4.2 indicates, the current version of DemandDriven Sales Manager assumes that the demand rate is going to be twice as high as the current level observed in the game unless there are less than 50 days left in the game. Apparently, this assumption is not always valid. A better approach

could be to estimate future demand based on information about the initial average number of customer RFQs and monitored estimated trend level. A cumulative demand over the number of days left in the game could then be calculated based on the results of a simulation of a random walk described in TAC-SCM game specification. This approach might better estimate the future cumulative demand since it could be randomly optimistic, pessimistic or realistic during the game. Apparently, more testing is needed to show any advantages of this proposed estimation technique over implemented one.

As it was explained in Chapter 3 and Chapter 4 both sales strategies use *OrderProbability* matrix to model the probability of order curve. However, in current strategy implementations these matrices keep track of different parameters that could influence the probability of order curve. Extensive analysis of several past games showed that there is a trade off between the size of *OrderProbability* matrix and its usefulness during the game. Apparently, the more parameters a strategy keeps track of, the larger the size of *OrderProbability* matrix. And the larger the size of *OrderProbability* matrix, the harder it is to adjust it during the game to increase the accuracy of prediction. As analysis of past games showed Demand-Driven Sales strategy keeps track of parameters that have more significant influence on the probability of order curve than those tracked by MaxEProfit Sales strategy. So, it could be said that DemandDriven way of modeling *OrderProbability* matrix is more efficient. Another difference is in the way how proposed strategies adjust *OrderProbability* matrix during the game. Apparently, more testing is needed to analyze the difference between proposed learning strategies and show its usefulness.

By analyzing the semi-finals games data, it could also be concluded that better integration and communication between the RM Inventories Manager and Sales Manager are needed. Currently the RM Inventory Manager tries to buy enough RM parts to keep producing FG products at the highest possible factory utilization rate - 100% during the game. The Sales Manager observes the current production rate as the game progresses and assumes that products will be produced at the same rate through the rest of the game. This assumption

eliminates the need of communication between these two components. However, if the RM Inventory Manager is not successful in getting enough RM parts to keep producing FG products at the desirable rate, the Sales Manager ends up selling out existing FG inventories cheaper than it could have sold if it knew ahead of time that there were not enough RM parts to keep producing products at the same rate. To solve this problem a better integration and communication between these two components might be explored in future versions of the agent. In other words, currently to predict the number of products the Sales Manager has to sell before the end of the game it uses the current moving average of the number of products produced and assumes that the production rate will be the same through the rest of the game.

$$FG\_products\_for\_sale = available\_FG\_inventory(product\_type) + \\ + products\_produced(product\_type) \times number\_of\_days\_left. \quad (6.1)$$

Apparently, the RM Inventory Manager has more information to estimate this number based on available and ordered RM parts. If the RM Inventory Manager provided this information, the Sales Manager wouldn't need to assume anything about the future production and could use more informed estimates provided by the RM Inventory Manager instead. On the other hand, the RM Inventory Manager could use current and expected profit data generated by the Sales Manager to decide if it is worthwhile to keep producing at the same rate. So, if the Sales Manager's data along with the RM Inventory Manager's data indicate that products are sold at a loss, an agent should try to lower the rate of acquiring new RM parts or stop doing so at all. In this case, the RM Inventory Manager's goal may be to keep the FG inventories at minimum levels until the Sales Manager's data indicate that agent is able to sell products profitably again.

As mentioned before, currently the RM Inventory Manager tries to buy enough RM

parts to keep producing FG products at as high as possible factory utilization rate, ideally at 100% rate. This strategy assumes that an average TAC-SCM game's limiting factor is factory utilization, which is not a valid assumption in the current game setup. An agent receives a significant RM parts discount by ordering on the first day. Ordering on the first day means that an agent doesn't have access to any information about the current game and has to decide blindly how many components to order on the first day. The current upfront assumption that every game has a RM supply bottleneck works perfectly well in supply-limited games when the agent is trying to build as many products as it possibly could and sells them at profit. However, as experiments showed, this strategy leads to huge losses in customer-demand-limited games due to a-priory over-commitment to buy many more RM parts than an agent could profitably sell during a game with even average customer demand. Hence, in the future, I think, it would be beneficial to perform a detailed analysis of many previous games with different customer demand levels and estimate how many products on average an agent could realistically expect to sell at profit in an average game. The goal of the RM Inventory Manager on the first day might then be to acquire enough RM parts to build products that MinneTAC could profitably sell in an average game. This number could realistically be expected to be less then the number of RM parts required to keep the factory utilization at 100% rate throughout the game. Apparently, an implementation of this idea would lead to lower scores during supply-limited games since MinneTAC will be forced to buy more RM parts at a full price, which would result in lower overall profit margins. On the other hand, the agent's flexibility to stop ordering RM supplies when it could not sell FG products profitably might be a positive result of this approach.

Appendix A

Simulation Results

The results of most representative TAC-SCM'03 games mentioned throughout the paper are presented here for consistency. The simulations were run on three publicly-accessible TAC-SCM servers:

1. http://kalamari.sics.se:8080/ configured for 56-day-long games for testing puproses.

2. http://tac5.sics.se:8080/ and http://tac6.sics.se:8080/ configured for 220-day games suggested by the TAC-SCM specification.

Brief game results summarized here along with the full game data useful for extended analysis could also be found at corresponding TAC-SCM server under the "Game History" link.

Player	Revenue	Interest	Costs	Penalty	PPer	PM1	PM2	Result
Botticelli	106.20	-0.63	75.42	1.25	2%	29%	27%	28.91
RedAgent	109.29	-0.69	77.93	2.64	3%	29%	26%	28.02
whitebear	67.28	-0.36	39.95	0.77	0%	41%	40%	26.88
UMBCTAC	81.63	-1.38	60.10	3.52	6%	26%	20%	16.64
MinneTAC	78.67	-0.91	74.64	0	0%	5%	4%	3.11
Sirish	23.61	-1.003	20.92	0.97	4%	11%	3%	0.72

Figure A.1: Result for game 605@tac5.sics.se played 2003-07-17 07:00:00. Summary of Income and Expenses. Each agent's mean revenue, interest, supply costs, penalties (both in millions of dollars and as a percent of a total revenue - PPer), profit margins (PM1 is the margin excluding bank interest and penalties while PM2 includes bank interest and penalties) and overall score are presented in this table. The data is in millions of dollars.

Player	Orders	Utilization	Deliveries(on-time/late/missed)	DPerf
Botticelli	6464	87%	6298 / 86 / 80	97%
RedAgent	6804	85%	6426 / 174 / 204	94%
whitebear	3992	48%	3941 / 50 / 1	99%
UMBCTAC	5041	66%	4325 / 664 / 52	86%
MinneTAC	5061	55%	5061 / 0 / 0	100%
Sirish	1889	23%	1802 / 22 / 65	95%

Figure A.2: Result for game 605@tac5.sics.se played 2003-07-17 07:00:00. Summary of delivery statistics. Each agent's mean number of orders, factory utilization, on-time deliveries, late deliveries, missed deliveries, and the overall delivery performance are presented in this table.

Player	Revenue	Interest	Costs	Penalty	PPer	PM1	PM2	Result
rudy0	93.09	-0.78	74.65	1.55	2%	20%	17%	16.11
PSUTAC	80.54	-1.53	67.88	0.96	1%	16%	13%	10.18
rudy	87.31	-0.44	82.27	0.70	1%	6%	4%	3.91
twoapple	4.72	-0.15	14.97	5.04	25%	-216%	-326%	-15.45
deepmaize	5.37	-0.16	15.71	5.02	24%	-191%	-288%	-15.51
lychee	52.77	-8.45	326.95	1.73	1%	-519%	-538%	-284.35

Figure A.3: Result for game 1180@tac5.sics.se played 2003-08-10 03:23:00. Summary of Income and Expenses. Each agent's mean revenue, interest, supply costs, penalties (both in millions of dollars and as a percent of a total revenue - PPer), profit margins (PM1 is the margin excluding bank interest and penalties while PM2 includes bank interest and penalties) and overall score are presented in this table. The data is in millions of dollars.

Player	Orders	Utilization	Deliveries(on-time/late/missed)	DPerf
rudy0	5034	63%	4855 / 45 / 134	96%
PSUTAC	6503	77%	6424 / 0 / 79	99%
rudy	5637	70%	5549 / 42 / 46	98%
twoapple	817	3%	292 / 49 / 476	36%
deepmaize	912	3%	279 / 87 / 546	31%
lychee	5248	89%	5100 / 0 / 148	97%

Figure A.4: Result for game 1180@tac5.sics.se played 2003-08-10 03:23:00. Summary of delivery statistics. Each agent's mean number of orders, factory utilization, on-time deliveries, late deliveries, missed deliveries, and the overall delivery performance are presented in this table.

Player	Revenue	Interest	Costs	Penalty	PPer	PM1	PM2	Result
lychee	61.88	-0.99	66.98	0.03	0%	-7%	-9%	-6.13
deepmaize	9.65	-0.40	22.50	1.25	5%	-132%	-149%	-14.50
twoapple	9.60	-0.42	22.26	1.68	7%	-131%	-153%	-14.75
PSUTAC	53.00	-1.74	68.11	0.06	0%	-28%	-31%	-16.91
rudy	48.43	-0.86	87.40	0.11	0%	-79%	-81%	-39.94
rudy0	26.63	-1.75	82.52	0.16	0%	-209%	-216%	-57.81

Figure A.5: Result for game 1181@tac5.sics.se played 2003-08-10 04:20:00. Summary of Income and Expenses. Each agent's mean revenue, interest, supply costs, penalties (both in millions of dollars and as a percent of a total revenue - PPer), profit margins (PM1 is the margin excluding bank interest and penalties while PM2 includes bank interest and penalties) and overall score are presented in this table. The data is in millions of dollars.

Player	Orders	Utilization	Deliveries(on-time/late/missed)	DPerf
lychee	5017	69%	5014 / 0 / 3	100%
deepmaize	1419	16%	1201 / 89 / 129	85%
twoapple	1381	16%	1125 / 104 / 152	81%
PSUTAC	3949	81%	3945 / 0 / 4	100%
rudy	3747	76%	3731 / 9 / 7	100%
rudy0	2004	73%	1978 / 20 / 6	99%

Figure A.6: Result for game 1181@tac5.sics.se played 2003-08-10 04:20:00. Summary of delivery statistics. Each agent's mean number of orders, factory utilization, on-time deliveries, late deliveries, missed deliveries, and the overall delivery performance are presented in this table.

Player	Revenue	Interest	Costs	Penalty	PPer	PM1	PM2	Result
whitebear	4.66	-0.77	30.00	0.00	0%	-542%	-559%	-26.11
Botticelli	22.37	-0.99	48.00	1.70	3%	-114%	-126%	-28.33
UMBCTAC	0.70	-1.27	37.54	0.00	0%	-5273%	-5455%	-38.12
TacTex	14.65	-2.25	58.68	1.48	2%	-300%	-325%	-47.76
Sirish	5.53	-1.78	55.69	0.05	0%	-906%	-939%	-51.98
MinneTAC	5.56	-3.10	75.82	0.25	0%	-1262%	-1322%	-73.61

Figure A.7: Result for game 1398@tac6.sics.se played 2003-08-12 14:00:00. Summary of Income and Expenses. Each agent's mean revenue, interest, supply costs, penalties (both in millions of dollars and as a percent of a total revenue - PPer), profit margins (PM1 is the margin excluding bank interest and penalties while PM2 includes bank interest and penalties) and overall score are presented in this table. The data is in millions of dollars.

Player	Orders	Utilization	Deliveries(on-time/late/missed)	DPerf
whitebear	387	26%	387 / 0 / 0	100%
Botticelli	4542	58%	4379 / 24 / 139	96%
UMBCTAC	424	35%	424 / 0 / 0	100%
TacTex	4275	55%	4037 / 151 / 87	94%
Sirish	1678	63%	1672 / 2 / 4	100%
MinneTAC	1534	62%	1491 / 16 / 27	97%

Figure A.8: Result for game 1398@tac6.sics.se played 2003-08-12 14:00:00. Summary of delivery statistics. Each agent's mean number of orders, factory utilization, on-time deliveries, late deliveries, missed deliveries, and the overall delivery performance are presented in this table.

Player	Revenue	Interest	Costs	Penalty	PPer	PM1	PM2	Result
Botticelli	13.14	-1.02	41.94	1.50	3%	-218%	-237%	-31.32
UMBCTAC	4.38	-1.07	41.81	0	0%	-854%	-878%	-38.50
TacTex	25.61	-1.59	61.66	5.53	8%	-140%	-168%	-43.17
whitebear	8.31	-1.10	55.73	0	0%	-570%	-583%	-48.52
MinneTAC	25.48	-1.89	74.42	0.50	1%	-191%	-200%	-51.33
Sirish	4.17	-1.20	54.59	0	0%	-1209%	-1238%	-51.62

Figure A.9: Result for game 1401@tac6.sics.se played 2003-08-12 17:00:00. Summary of Income and Expenses. Each agent's mean revenue, interest, supply costs, penalties (both in millions of dollars and as a percent of a total revenue - PPer), profit margins (PM1 is the margin excluding bank interest and penalties while PM2 includes bank interest and penalties) and overall score are presented in this table. The data is in millions of dollars.

Player	Orders	Utilization	Deliveries(on-time/late/missed)	DPerf
Botticelli	3176	41%	3034 / 34 / 108	96%
UMBCTAC	1338	41%	1338 / 0 / 0	100%
TacTex	3918	56%	3149 / 86 / 683	80%
whitebear	862	49%	862 / 0 / 0	100%
MinneTAC	2760	79%	2699 / 6 / 55	98%
Sirish	1003	53%	1003 / 0 / 0	100%

Figure A.10: Result for game 1401@tac6.sics.se played 2003-08-12 17:00:00. Summary of delivery statistics. Each agent's mean number of orders, factory utilization, on-time deliveries, late deliveries, missed deliveries, and the overall delivery performance are presented in this table.

Bibliography

[1] R. Arunachalam, J. Eriksson, N. Finne, S. Janson, and N. Sadeh. The TAC supply chain management game (Draft version 0.6), March 2003.

[2] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa. RoboCup: The robot world cup initiative. In W. L. Johnson and B. Hayes-Roth, editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 340–347, New York, 1997. ACM Press.

[3] H. Kitano, S. Tadokor, H. Noda, I. Matsubara, T. Takhasi, A. Shinjou, and S. Shimada. Robocup-rescue: Search and rescue for large scale disasters as a domain for multi-agent research. In *Proceedings of the IEEE Conference on Systems, Men, and Cybernetics*, 1999.

[4] C. McMillen. Toward the development of an intelligent agent for the supply chain management game of the 2003 trading agent competition, May 2003.

[5] P. Stone. Multiagent competitions and research: Lessons from RoboCup and TAC. In *The RoboCup 2002 International Symposium*, 2002.

[6] P. Stone and A. Greenwald. The first international trading agent competition: autonomous bidding agents. *Electronic Commerce Research*, 2002.