

MinneTAC Sales Strategies for Supply Chain TAC

Wolfgang Ketter, Elena Kryzhnyaya, Steven Damer, Colin McMillen,
Amrudin Agovic, John Collins, and Maria Gini*
Dept. of Computer Science and Engineering, University of Minnesota

Abstract

We describe two sales strategies used by our agent, *MinneTAC*, for the 2003 Supply Chain Management Trading Agent Competition (TAC SCM). Both strategies estimate, as the game progresses, the probability of receiving a customer order for different prices and compute the expected profit. We empirically analyze the effect of the discount given by suppliers on orders made the first day of the game, and show that in high-demand games there is a strong correlation between the performance of an agent in the game and the offers it receives from suppliers the first day of the game.

1. MinneTAC Sales Strategies

In TAC SCM [6] six autonomous agents compete to maximize profits in a computer-assembly scenario. Agents earn money by selling computers they assemble out of parts purchased from suppliers. The agent with the highest bank balance at the end of the game wins. To obtain parts, an agent must send a *request for quotes* (RFQ) to an appropriate *supplier*. Suppliers respond either by specifying the price for the supplies, or, if they cannot satisfy the request in full, by giving a partial offer with a revised quantity and price, or the earliest date the request can be satisfied in full. Every day each agent receives a set of RFQs from *customers* requesting to buy computers. Customers accept the lowest bid which is at or below their reserve price.

Prior to the competition, we analyzed the game and determined that the factor more likely to limit the production of PCs on any day is lack of supplies. Hence, we decided to use a *supply-driven* sales strategy. We developed two variants, that we call *MaxEProfit* and *DemandDriven* [4]. In both strategies, each day the agent determines the offer price for each RFQ and sorts its offers by decreasing profit margin, where $ProfitMargin = (price - cost)/price$. To determine how many offers to make, the agent considers its

uncommitted finished goods inventory. For each offer it reserves a fraction of computers of that type according to the estimated probability of receiving an order $P(order)$.

MaxEProfit determines the offer price to maximize its expected profit margin from each order, with the constraint that the price has to be greater or equal a target price for the product. The target price is updated during the game to reflect current market prices for each computer type. Since $P(order)$ is estimated from many parameters, often there are not enough data until very well into the game for the estimates to be accurate. This observation lead us to design the *DemandDriven* strategy.

DemandDriven sets a target probability for receiving an order, $TargetProb$, and determines the offer price accordingly, using the reverse cumulative density function of $P(order)$. The objective is to sell out the inventory by the end of the game. The strategy starts out by computing $P(order)$ from past data. The probabilities don't change through the game, but the entire curve is shifted when there is a mismatch between the orders accepted and $TargetProb$. If more orders are accepted than expected, the prices are increased, if fewer are accepted prices are decreased.

In Figure 1 we show the market reports for computers of type 4 during two different games and the prices predicted using *MaxEProfit* and using *DemandDriven*. The top two figures are for a high-demand game, the last one for a low-demand. We can observe how the predicted prices match actual product prices. In a low-demand situation, the competition is very high and therefore the prices are lower.

2. Analysis of Start-Effect

We analyzed many games to see how the large discount given by suppliers on orders made the first day, coupled with the random sequence in which agent RFQs are considered, affects the outcome of the game. We developed a measure, called *delay measure* (DM), which measures for each agent the delay in receiving the components ordered the first day weighted by the order total value: $DM = \frac{\sum_{i=1}^{\#RFQs} Value(RFQ_i) \times DueDate(Offer_i)}{\sum_{i=1}^{\#RFQs} Value(RFQ_i)}$

where $Value(RFQ_i)$ is computed using the components

* Partial support research is gratefully acknowledged from the National Science Foundation under award NSF/IIS-0084202.

base price, and *DueDate* is the due date of the offer. We applied the measure to games played at the 2003 ICEC (Pittsburgh, Oct. 03). In high-demand games we calculated a correlation coefficient of 0.5381 between bank status and the amount ordered on the first day, and of $-.03456$ between bank status and DM. In low-demand games the correlation coefficient between bank status and amount ordered on the first day was -0.3242 , between bank status and DM was 0.3904. This indicates that agents who did not order many parts did better in low-demand games.

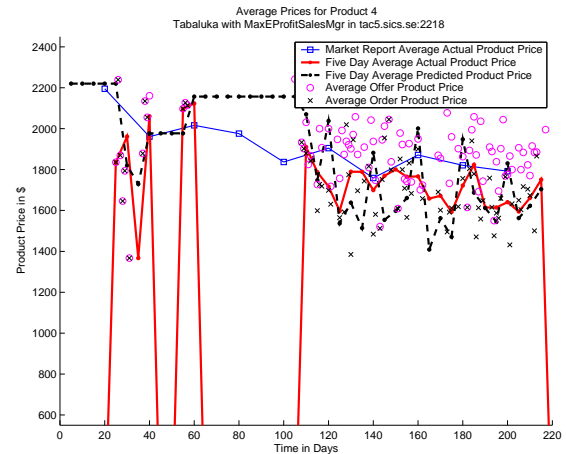
3. Related Work and Conclusions

Nearly all agents in last year's competition used some way of modeling the probability of receiving an order. Botticelli [1] uses a linear cumulative density function (CDF) to determine the relationship between offer price and order probability. We use a reverse CDF and take other factors into account, such as quantity and due date. TacTex [5] uses the lowest and highest offer price, which are provided for each product every day by the game server, and determines the probability of an order by linear interpolation. RedAgent [3], uses an internal marketplace structure with competing bidders to set offer prices. PackaTAC [2] lets other agents set the price and tries to follow.

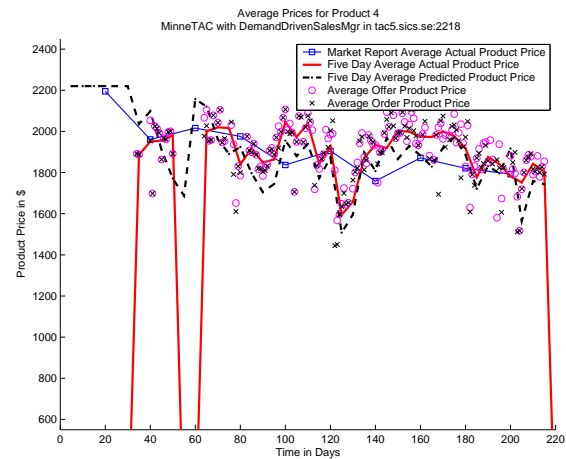
Since we estimated the bottleneck was going to be in the supply and not in the production, we did not worry, as other teams [1, 5], about optimizing the production of our agent. Both MinneTAC's sales strategies were competitive in high-demand games, but often sold larger quantities at lower margins compared to other agents. DemandDriven performed better in low-demand games, but was unable to compensate for the large number of parts ordered early in the game under the assumption of a supplier capacity bottleneck.

References

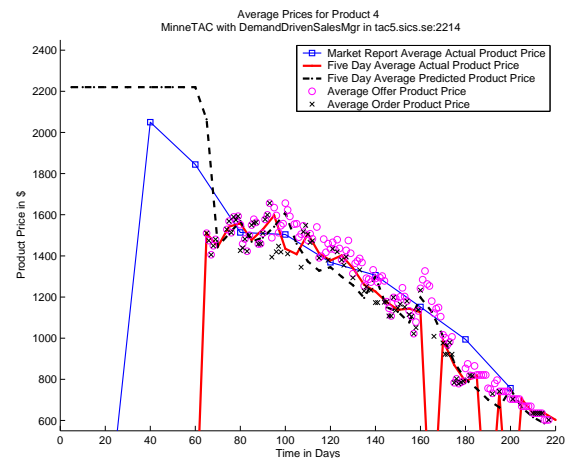
- [1] M. Benisch, A. Greenwald, I. Grypari, R. Lederman, V. Naroditskiy, and M. Tschantz. Botticelli: A supply chain management agent designed to optimize under uncertainty. *ACM Trans. on Computational Logic*, 4(3):29–37, 2004.
- [2] E. Dahlgren and P. Wurman. PackaTAC: A conservative trading agent. *SIGecom Exchanges*, 4(3):33–40, 2004.
- [3] P. W. Keller, F.-O. Duguay, and D. Precup. Redagent - winner of the TAC SCM 2003. *SIGecom Exchanges*, 4(3):1–8, 2004.
- [4] W. Ketter, E. Kryzhnyaya, S. Damer, C. McMillen, A. Agovic, J. Collins, and M. Gini. Design and analysis of the MinneTAC-03 Supply-Chain Trading Agent. Technical Report 04-016, University of Minnesota, Dept of Computer Science and Engineering, Minneapolis, MN, 2004.
- [5] D. Pardoe and P. Stone. TacTex-03: A supply chain management agent. *SIGecom Exchanges*, 4(3):19–28, 2004.
- [6] N. Sadeh, R. Arunachalam, J. Eriksson, N. Finne, and S. Janson. TAC-03: A supply-chain trading competition. *AI Magazine*, 24(1):92–94, 2003.



Game 2218 – MaxEProfit



Game 2218 – DemandDriven



Game 2214 – DemandDriven

Figure 1. Timeline in a high and low-demand game. Circles indicate average offer prices the agent made and crosses average order prices received.