Evaluating Adaptive Customer Strategies in TAC SCM

Yevgeniya Kovalchuk, Maria Fasli

Department of Computing and Electronic Systems, University of Essex
Wivenhoe Park, Colchester, CO4 3SQ, UK
{yvkova; mfasli}@essex.ac.uk

Abstract

Supply Chain Management (SCM) is a complex process which includes a number of interrelated activities such as: negotiating with suppliers for raw materials, competing for customer orders, managing inventory, scheduling production, and delivering goods to customers. In this paper we present a number of strategies to be examined in the domain of SCM. We introduce a multi-agent system which we used to evaluate the proposed methods. We tested the system in the Trading Agent Competition SCM game, which offers a realistic simulated environment for studying SCM strategies. Although we introduce a number of strategies, we concentrate on the ones for predicting winning bidding customer prices to support a successful performance on the customer side of the supply chain.

1 Introduction

In today's highly dynamic, time-constrained environments, developing efficient decision support systems is a key challenge. In particular, in the domain of SCM, which deals with the planning and coordination of the activities of organizations from getting raw materials, manufacturing goods to delivering them to customers, supporting dynamic strategies is a major but unresolved issue. All entities in the supply chain are highly connected and interdependent. Being successful in one area of the supply chain does not necessarily guarantee the improvement of the overall performance. Thus, there is the need for a mechanism to separate different tasks and explore them both independently and in relation to each other. We implemented such a mechanism in our multi-agent decision support system which we tested in the TAC SCM game (Collins et al. 2006). Using a multi-agent approach, we built a number of TAC SCM agents and allowed them to compete against each other in order to compare the performance of each proposed algorithm. Although we discuss a number of strategies in the paper, we mainly concentrate on those that deal with the problem of predicting customer offer prices that could result in customer orders (winning bidding prices).

The TAC SCM participants take different approaches in deciding the price to be offered to customers in response to their requests for quotes (RFQs) (Pardoe and Stone 2006, Benish, Andrews, and Sadeh 2006, He et al. 2006, Kontogounis et al. 2006, Ketter et al. 2004, Keller, Dugay, and Precup 2004, Dahlgren and Wurman 2004). In this paper we propose two different approaches for predicting winning bidding prices. The first is to predict such prices based on the RFQ details and current market situation. The second one is to predict the competitors' bidding prices and bid just below the minimum predicted value. To the best of our knowledge, the second proposed strategy has not yet been explored within the TAC community. We used the Neural Networks learning technique to complete both tasks.

The rest of the paper is organized as follows. First, we introduce our multi-agent system Socrates and describe a number of algorithms for the SCM implemented in the system. The experiment settings are provided next followed by a discussion of the results obtained. The paper closes with the conclusions and a discussion of future work.

2 The Design of Socrates

For our SCM system we chose a multi-agent approach which allows us to break down the whole system into separate building blocks, each concentrating on a particular part of the supply chain. By replacing one building block with another and by combining them in different ways, we create various versions of our own system in order to check how different strategies influence the overall system's performance. The system includes the following agents: Manager, Demand Agent, Supply Agent, Inventory Agent, Production Agent, and Delivery Agent. The Manager agent is responsible for the communication with the TAC server as well as managing all other agents. The Demand Agent decides which customer RFQs to answer and with what price. The remit of the Supply Agent is the procurement of low cost components on time from suppliers. The Inventory Agent manages the component and PCs stocks in order to satisfy the needs of the Production and Delivery agents while at the same time minimising holding costs. The Production Agent is responsible for scheduling current production and projecting production for the future. Finally, the Delivery Agent deals with delivering PCs to customers according to their orders and on time to prevent penalties.

Model name	Number of Units		Attributes	Price range
	Input Layer	Hidden Layer		
Short fixed NN	5	3	current date, penalty per unit, reserve price, latest reported lowest order price, boolean indicator of whether an offer became an order	Fixed
Full fixed NN	13	7	current date, quantity, due lead time, penalty per unit, reserve price, lowest order price reported for the last 3 days, highest order price reported for the last 3 days, order level=[order number/RFQ number], boolean indicator of whether an offer became an order	Fixed
Full dynamic NN	13	7	current date, quantity, due lead time, penalty per unit, reserve price, lowest order price reported for the last 3 days, highest order price reported for the last 3 days, order level=[order number/RFQ number], boolean indicator of whether an offer became an order	Varied

Table 1. Models for predicting own winning bidding prices.

3 Predictive Strategies for Setting Customer Offer Prices

Although we implemented a number of strategies in the scope of each agent, we mainly concentrate on the ones for the Demand Agent. In particular, we propose two strategies for determining bidding prices which may result in customer orders (winning bidding prices) – in the TAC SCM game, customers award an order to the manufacturer which offers the lowest price among all competitors. The first strategy is to predict the probability of the winning price for each RFQ to be within a particular price range. We then set our offer price to the average of the price interval with the highest predicted probability. The second strategy is to predict the competitors' bidding prices and bid just below the minimum predicted value.

We applied the Neural Networks learning technique for both strategies using the Back-propagation algorithm to train them on the data from the log-files of previously played games. According to the first strategy, 16 ensembles of feed-forward neural networks (NN), one for each PC type, predict probabilities of winning a customer order for the specified PC type depending on the offer price set. Each ensemble is responsible for a particular price interval and outputs the probability of the price within this interval being accepted by a customer. We developed and tested 3 different NN architectures, which are summarized in Table 1. Inputs were normalized using the corresponding minimum and maximum allowed values for each input. For setting minimum and maximum values of the price units, we took two different approaches. Two of our NN models have these values fixed and set to the minimum and maximum values observed in the games. The third NN model sets these values dynamically for each RFQ separately. More specifically, the minimum value is determined as the discounted latest lowest order price stated in the price report for the respective PC type: [LowestOrderPrice - 200]. The maximum value is set to [ReservePrice + 50].

For our second strategy, we built two kinds of NNs for every competitor, each predicting the competitor's offer price for the given RFQ. For the first type of NNs (Identical NN Competitor Bid Price Predictors), we used the same set of inputs for each competitor which includes 13 attributes, namely: PC type, current date, lead time, quantity, reserve price, penalty, the lowest and the highest market price reported during the last 3 days, and current demand level. For the second type of NNs (Individual NN Competitor Bid Price Predictors), we used a separate set of inputs, depending on the parameters that each competitor considers while deciding on its bidding prices. To test the latter type of NNs, we set up all the competitors so that the information on the parameters they use for setting their offer prices is known. However, we make no assumptions on how these parameters are used by competitors; the predictors have to evolve the models of the competitors' behavior.

4 Experiments

In order to determine how separate algorithms contribute to the overall performance of the SCM system designed for the TAC SCM game, we created a number of Socrates agents, each following different strategies. In order to compare the algorithms' performance in the context of the same market settings, we allow different versions of the agent to compete against each other, rather than against the agents developed by other TAC SCM participants. By running experiments we aim not only to develop the most successful strategy for the agent to compete in the TAC SCM game, but also to check the following hypotheses:

1. The agent that plans its scheduling in advance performs better than the one that does not.
2. The agent that tracks the supplier market performs better than the one that does not.
3. The agent that bids only on profitable customer RFQs performs better than the one that does not estimate potential profit of these RFQs.
4. The agent that tries to predict winning bidding prices for customer RFQs performs better than the one that does not.

5. The agent that tries to predict its competitors' bidding prices performs better than the one that tries to predict winning bidding prices based only on bidding history.
6. The model for predicting one's winning bidding prices that considers more attributes from the environment performs better than the one with a few inputs.
7. Dynamic restriction of the allowed price range for normalizing price inputs in the model for predicting one's winning bidding prices improves the performance of the model.
8. The agent that knows which parameters its competitors use for deciding on bids performs better than the agent that does not have such knowledge.
9. The Competitor Predicting models perform better for competitors that follow the determined bidding strategy than for competitors that bid randomly.
10. The Identical NN Competitor Bid Price Predictor model performs better than the Individual NN Competitor Bid Price Predictor model in the circumstances when a competitor bids randomly.

For each hypothesis we played 15 games; this appears to be enough (in terms of the level of results and their standard deviation) to evaluate the hypothesis. To get data to train our predictive models we played another 50 games. The competing agents were designed as follows.

For hypothesis 1, 2, and 3:

- Socrates – tracks the supplier market, chooses the lowest suppliers, and sets its reserve prices for components accordingly. The agent plans its production 12 days ahead and sets its offer prices to a random value between the lowest and highest market prices reported for the respective PC type the day before.
- Socrates1 – tracks the supplier market, bids on random customer RFQs and sets customer offer prices to [ReservePrice * (1.0 - random.nextDouble() * 0.2))].
- Socrates2 – does not track the supplier market, bids on random customer RFQs and sets customer offer prices to [ReservePrice - penalty/quantity].
- Socrates3 – plans production ahead as Socrates does, but doesn't track the supplier market. It bids on random customer RFQs with the offer price set to the average between the latest lowest and highest reported prices.
- Socrates4 – the agent is set similarly to Socrates3 with the only difference that the agent sets its customer offer prices as Socrates1 does, i.e. to [ReservePrice * (1.0 - random.nextDouble()*0.2))].
- Socrates5 – the agent is set similarly to Socrates3 with the only difference that the agent sets its customer offer prices as Socrates does, i.e. randomly between the lowest and highest market prices reported for the corresponding PC type on the previous day.

For hypothesis 4-8 agents differ only in the way they set customer offer prices:

- Socrates – sets prices according to the lowest competitor price among those predicted for all the competitors for a given RFQ. Both NN models described in section 3 are tested simultaneously in order to have the same base for comparison of the algorithms. Offer price is set according to the Individual NN model.
- Socrates1 – sets prices according to the predictions of the Short Fixed NN model (section 3).
- Socrates2 – sets prices to [reserve price - penalty/quantity].
- Socrates3 – sets prices to the average between the lowest and highest reported.
- Socrates4 – sets prices according to the predictions of the Full Fixed NN model (section 3).
- Socrates5 – sets prices according to the predictions of the Full Dynamic NN model (section 3).

We fixed all other strategies for all the competing agents in order to get a clearer picture of how bidding strategies influence the agents' performance. The agents do not track supplier market, schedule production for 12 days in advance, consider customer RFQs in profit descending order, and deliver produced PCs as soon as they are available to satisfy all active orders sorted by due date.

For hypothesis 9 and 10, we replaced Socrates5 with an agent that sets customer offer prices randomly between the lowest reported and reserve prices.

5 Results

Our results demonstrate that the agents that track the supplier market, plan their production in advance and/or pick only profitable customer RFQs, perform better than those that do not support these strategies, which proves hypotheses 1, 2 and 3. The most successful agent is the one that uses all these three strategies. Regarding the performance on the Demand part, we discovered that rather than following the static strategy of setting customer offer prices based only on two parameters, a more sophisticated strategy should be implemented.

The agents designed to check hypothesis 4-8, paid a similar rate of component prices and were able to deliver customer orders on time, which results in low penalty costs. Thus, the scores of the agents depend only on the number of orders they get from customers and on the profitability of these orders. This allows us to evaluate the effectiveness of the bidding strategies (section 3) by comparing the agents' (a) levels of customer offer prices; (b) overall scores; and (c) order winning rates (the ratio between the number of offers send to the number of orders received).

The agents that predict their own winning bidding prices show relatively high results, which supports hypothesis 4. The agent that incorporates the Full Dynamic NN algorithm performed slightly better, which proves our hypothesis 7. We did not find strong evidence for our hypothesis 6 however: although the Full Fixed NN model provides higher winning rate than the Short Fixed Model, the overall score of the agent that applies the first model is much lower than the score of the agent that uses the second predictive model. The reason for this is that the Full Fixed NN model predicts on average lower price values than the Short Fixed Model.

As suggested by our hypothesis 5, both the winning rate and the overall score of the agent that predicts the competitors' bidding prices indicate that this agent is the most successful. The offer prices for all PC types set by this agent are on average just bellow those of all other agents, which demonstrates the effectiveness of the strategy to bid according to the predicted competitors' prices, and the power of NNs as a learning technique for solving this particular task. At the same time, the analysis of individual cases from the games played reveals that the discussed agent sometimes bids lower than it possibly should. This can be explained by the following: if the agent with the lowest predicted price does not bid for an RFQ, then the winning price will be the lowest among the ones set by the other agents who actually bid. This suggests the need to implement classifiers for each agent, in addition to the predictors of bidding prices, which would indicate whether the agent is actually going to bid for an RFQ, given the predicted price for this RFQ.

We compare the performance of our two Competitor Bid Price Predictor models in terms of the accuracy of their predictions. In particular, we calculated the root mean square error (RMSE) using the normalised actual and predicted price values observed in all games. According to the results provided in Table 2, the following conclusions can be drawn: (1) the more complicated strategy a competitor is taking and the more attributes it considers for making its bidding decisions, the harder it is to predict its bidding prices; (2) the Individual Competitor Bid Price Predictors, that use in their models only those attributes which competitors consider, perform better than the Identical Competitor Bid Price Predictors that use the full set of attributes for all the competitors (which proves our hypothesis 8); (3) both types of predictors have relatively high prediction error for the agent that bids randomly between the lowest reported and reserve prices compared to all other agents (which supports our hypothesis 9). We did not find the evidence for hypothesis 10, as both predictive models show similar error rate for the agent that bids randomly.

Algorithm	Socr.1	Socr.2	Socr.3	Socr.4	Socr.5
Socrates5 follows the Full Dynamic NN model					
Individ. NN	0.0194	0.0038	0.0033	0.0141	0.0178
Identic. NN	0.0203	0.0279	0.0067	0.0243	0.0216
Soc.5 bids randomly between the lowest reported and reserve prices					
Individ. NN	0.0176	0.0042	0.0053	0.0116	0.0602
Identic. NN	0.0259	0.0278	0.0085	0.0310	0.0608

Table 2. Prediction RMSE for the Individual and Identical NN Competitor Bid Price models.

6 Conclusions and Future Work

In this paper we presented a multi-agent system for SCM. The multi-agent approach gives the opportunity to break the complex domain into simpler building blocks. By replacing one building block with another, we built a number of SCM agents who followed different strategies. We let these agents compete against each other in the TAC SCM game, and the results from the games demonstrate the contribution of separate strategies into the overall success of competing agents. In particular, we proved the importance of tracking the supplier market, projecting production in advance and estimating potential profit of customer RFQs. Our major contribution in this paper however is related to the problem of predicting customer winning bidding prices. We developed two different approaches for solving the task. The first one is to predict the prices based on the information perceived form the environment. The second one is to predict the competitors' bidding prices. The Neural Network learning technique was applied to both tasks. While both predictive approaches outperformed other robust algorithms presented in the paper, modelling the competitors' strategies showed to be the most powerful technique for environments which have a game format (i.e. a number of participants compete against each other in order to get the best score). We have found, however, that the prediction of the competitors' bidding prices themselves is not enough for making optimal decisions on offer prices: if the agent with the lowest predicted price does not bid for an RFQ, then the winning price will be the lowest among the ones set by the other agents who actually bid. Thus, in addition to the prediction of the agents' bidding prices for every RFQ, as part of future work we are going to implement classifiers that will specify whether the agent will actually bid for the RFQ at such price level.

Our experiments also showed that knowledge of the features that the competitors are using for making their decisions, could improve the predictive models of these competitors. As this knowledge is not usually available in advance in such domains, our task now is to derive it by observing the competitors' performance and perceiving information from the environment.

In this paper we tested different versions of only one agent in order to derive the most successful strategies for this agent to follow. To check the robustness of the proposed algorithms for predicting customer winning bidding prices, games against the agents proposed by other TAC SCM participants have to be played additionally.

References

Benish M., Andrews, J., and Sadeh, N. 2006. Pricing for Customers with Probabilistic Valuations as a Continuous Knapsack Problem. In Proceedings of the Eighth International Conference on Electronic Commerce , 38-46. Fredericton, Canada: ICEC-06.

Collins, J., Arunachalam, R., Sadeh, N., Eriksson, J., Finne, N., and Janson, S. 2006. The Supply Chain Management Game for the 2007 Trading Agent Competition, Technical Report CMU-ISRI-07-100, Carnegie Mellon University.

Dahlgren, E. and Wurman, P. 2004. PackaTAC: A conservative trading agent. *ACM SIGecom Exchanges* 4(3): 38-45. New York, NY, USA: ACM Press.

He, M., Rogers, A., Luo, X., and Jennings, N.R. 2006. Designing a Successful Trading Agent for Supply Chain Management. In Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems, 1159-1166. Hakodate, Japan: AAMAS-06.

Keller, W., Dugay, F.-O., and Precup, D. 2004. RedAgent – winner of the TAC SCM 2003. *ACM SIGecom Exchanges* 4(3): 1-8. New York, NY, USA: ACM Press.

Ketter, W., Kryznaya, E., Damer, S., McMillen, C., Agovic, A., Collins, J., and Gini, M. 2004. MinneTAC sales strategies for supply chain TAC. In Proceedings of the Third International Conference on Autonomous Agents and Multi-Agent Systems , 1372-1373. New York, NY: AAMAS-04.

Kontogounis, I., Chatzidimitriou, K.C., Symeonidis, A.L., and Mitkas, P.A. 2006. A Robust Agent Design for Dynamic SCM Environments. In Proceedings of the Fourth Hellenic Joint Conference on Artificial Intelligence, 18-20. Heraklion, Greece: SETN'06.

Pardoe, D. and Stone, P. 2006. Predictive Planning for Supply Chain Management. In Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling, 21-30. Cumbria, UK: ICAPS-06.