

The Trading Agent Competition as a test problem for Constraint Solving under Change and Uncertainty

Kenneth N. Brown, David A. Burke, Brahim Hnich, Onur Koyuncu and Armagan Tarim

Cork Constraint Computation Centre, Department of Computer Science,
University College Cork, Ireland
k.brown@cs.ucc.ie

1 Introduction

The annual Trading Agent Competition includes a Supply Chain Management game [1], in which agents must win contracts, obtain supplies, and produce goods, in competition with five other agents. In this position paper, we outline the problem domain, explaining why we believe it is a suitable testbed for reasoning about change and uncertainty in constraint optimisation, and we describe our initial attempts to design and implement a constraint-based agent to participate in the competition.

2 TAC SCM

The Trading Agent Competition Supply Chain Management games simulates a dynamic competitive supply and production environment. Multiple customers issue requests for quotes for the sale of PCs, including a quantity, a due date, and a lateness penalty. A number of production agents make bids for the contract, and for each request, a winning bid is selected. The winning agent must then manufacture the PCs, or provide them from stock, and ship them to the customer by the due date. In order to manufacture the PCs, the agent must procure raw material from suppliers. Again, requests for quotes (including delivery dates) are issued by the agent, the suppliers respond with offers, and an appropriate offer is selected. Each agent is limited by the capacity of its assembly line, and must select each day which set of products are to be manufactured. An agent has unlimited storage capacity, but must pay an inventory holding cost for each component and each finished PC. Failure to meet a due date on an order results in financial penalties. Each agent has an unlimited overdraft, but must pay interest at a rate higher than that received for a positive balance. The game consists of a year of 220 trading days, and each simulated day lasts for 15 seconds in real time. The aim of the game is to have the largest bank balance at the end of the year. The simulation is implemented in Java, and is controlled from a central server, to which remote agents must be connected. Basic Java agents are provided for

initial use. The competition consists of several rounds, and each round consists of many individual games. The participants in each round are ranked by their average closing bank balance.

The game clearly involves resource management, it is dynamic, and it involves significant uncertainty. There are in total 16 different PC configurations possible, each with a different set of components, and each requiring a different number of production cycles. The configurations are divided into three categories (*low, medium and high*) based on their expected price. Each day, the agent must decide which production cycles to devote to producing which configurations, based on the current set of actual and expected orders, and on the states of component and finished-goods inventory. Customer demand is uncertain - each day, the demand for configurations in each of the three categories varies based on a poisson distribution around a target which is varied as a random walk. The due dates and penalties (and maximum prices) are selected uniformly at random over an interval. Thus when scheduling production, an agent cannot know with certainty how many future orders will be available. Further, each day the agent makes offers, but does not receive results until the following day. The success rate will depend on the bids of the other agents. Thus each day the agent has an upper bound on the number of contracts which may be received tomorrow, but the actual contracts are not known with certainty until the following day. Daily reports on the maximum and minimum prices paid for each configuration are available. Every twenty days, the agent receives a market summary report, detailing the average prices. There is also uncertainty on the supplier side. An agent issues requests for the supply of components, and on the following day learns what offers have been made by the suppliers. The suppliers have limited resources, and will quote prices based on capacity, demand and the agent's reputation, and hence price depends both on random variables and on the behaviour of the other competing agents. Finally, the actual production capacity of the suppliers varies under a random walk from day to day, and so it is possible that the suppliers over-commit and deliver the components late.

Although the underlying constrained resource allocation problem is relatively simple, it is an online problem with uncertainty in the availability of resources. Further, the agents have some ability to make decisions on how the problem should change over time (by choosing which contracts to initiate), but that process is subject to significant uncertainty, caused partly by the game parameters, and partly by the presence of other agents acting in the game. Effective resource management therefore requires us to reason about both the changes and the uncertainty, and the competition provides a real and effective testbed.

3 Designing a constraint-based agent

We have designed and implemented an agent using Java and OPL, to compete in TAC-SCM 2005. Each trading day, we break the problem down into three decisions: (i) which PCs to manufacture, (ii) which components to request and order, and (iii) which PC orders to bid for and at what price. For (i), we are

initially taking a simple approach. We only schedule orders which have been confirmed, and each day, from the confirmed orders, we select jobs in order of due date that have the necessary components in stock, and schedule them until the production cycles are used up. For (ii), we attempt to order all components in advance. We maintain an expected order rate for each category of PC, which is updated each day based on the history of orders. Based on the quotes we received on the previous day, we make orders each day with long due dates to bring the component stocks up to the level required for the expected finished goods orders. In addition, we make short due date purchases to correct any errors in our previous estimates for both customer orders and supplier lead-times - that is, to bring stocks up to the levels required to fulfill confirmed orders.

Most of our effort has focused on deciding which bids to make each day, and on what prices to offer. We represent this as a constraint optimisation problem subject to uncertainty. Our aim is to select bids and prices to maximise our expected revenue, given component, inventory, lateness and interest costs, and given constraints on component availability and production capacity. However the success of any given bid depends on the actions of the other agents, and we are not given detailed information on their behaviour - we see only the minimum and maximum order prices for the previous day, and every twenty days we see the average order price for that period. Rather than try to reason explicitly about the behaviour of the other agents, we model the market conditions for each product as random variables, and we try to learn the probability of a bid price being accepted. We then use these probabilities in our optimisation model, and use expected production requirements and expected profits. Our aim is then to choose the bids and prices that maximise our expected profits, while ensuring that our expected demand does not exceed our capacity or supplies. It is possible that we succeed on too many bids, and then our production scheduling must decide which orders to satisfy.

This model is implemented in OPL Studio, and is embedded inside a Java agent. Each day, the agent updates its probability estimates for the success of different bids, its inventory, and its models of the suppliers; decides upon the day's production and delivery schedule; reads the RFQs from customers; runs the OPL model to make the bids; orders components from suppliers; and finally issues requests for quotes to suppliers for tomorrow's orders. The agent is currently participating in the competition, and appears to be robust and profitable, although not one of the leading agents. Future work will concentrate on improving the production scheduling and the management of supplies.

4 Conclusion

The Trading Agent Competition Supply-Chain Management game is a real time simulation of a competitive, dynamic and uncertain marketplace. As such, it provides an effective testbed for practical implementations of constraint solving under uncertainty. We have outlined our constraint-based agent for participat-

ing in the competition, and we will report on its final performance during the workshp.

5 Acknowledgments

This work is supported by Science Foundation Ireland under Grant 03/CE3/I405 as part of the Centre for Telecommunications Value-Chain-Driven Research (CTVR).

References

1. J. Collins, R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne and Sverker Janson "The Supply Chain Management Game for the 2005 Trading Agent Competition", Tech report CMU-ISRI-04-139, School of Computer Science, Carnegie Mellon University, 2004..