# CMieux: Adaptive Strategies for Competitive Supply Chain Trading

Michael Benisch, Alberto Sardinha, James Andrews and Norman Sadeh
School of Computer Science, Carnegie Mellon University

Existing supply chain management practices consist primarily of static interactions between established partners. Global competition, shorter product life cycles and the emergence of Internet-mediated business solutions create an incentive for exploring more dynamic supply chain practices. The Supply Chain Trading Agent Competition (TAC SCM) was designed to explore approaches to dynamic supply chain trading. TAC SCM pits against one another trading agents developed by teams from around the world. This paper presents Carnegie Mellon University's 2005 TAC SCM entry, the CMieux supply chain trading agent. CMieux implements a novel approach to coordinating supply chain bidding, procurement and planning, with an emphasis on the ability to rapidly adapt to changing market conditions. We present empirical results based on 200 games involving agents entered by 25 different teams during what can be seen as the most competitive phase of the 2005 tournament. Not only did CMieux perform among the top five agents, it significantly outperformed these agents in procurement while matching their bidding performance.

Categories and Subject Descriptors: I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems; I.6.0 [**Simulations and Modelling**]: General

General Terms: Multi-agent Systems

Additional Key Words and Phrases: Trading agents, Supply chain management, TAC SCM

## 1. INTRODUCTION

Existing supply chain management practices consist primarily of static interactions between established partners [Chopra and Meindl 2004]. As the Internet helps mediate an increasing number of supply chain transactions, there is a growing interest in investigating the potential for exploring the benefits of more dynamic supply chain practices [Arunachalam and Sadeh 2005; Sadeh et al. 1999]. The Supply Chain Trading Agent Competition (TAC SCM) was designed to explore approaches to dynamic supply chain trading. TAC SCM pits against one another trading agents developed by teams from around the world. Each agent is responsible for running the procurement, planning and bidding operations of a PC assembly company, while competing with others for both customer orders and supplies under varying market conditions.

This paper presents Carnegie Mellon University's 2005 TAC SCM entry, the CMieux supply chain trading agent. CMieux's architecture departs markedly from traditional Enterprise Resource Planning architectures and commercially-available supply chain management solutions through its emphasis on tight coordination between supply chain bidding, procurement and planning. Through this coordination, our trading agent is capable of adapting rapdily to changing market conditions and outperform its competitors. In particular, we present empirical results based on 200 games involving agents entered by 25 different teams during what can be seen as the most competitive phase of the 2005 tournament. Not only did CMieux perform among the top five agents, it significantly outperformed these agents in procurement while matching their bidding performance.

## 2. TAC SUPPLY CHAIN MANAGEMENT

This section provides a brief summary of the TAC Supply Chain Management game. The full description can be found in the official specification document [Collins et al. 2005].

The TAC SCM game is a simulation of a supply chain where six computer manufacturer agents compete with each other for both customer orders and components from suppliers. A server simulates the customers and suppliers, and provides banking, production, and warehousing services to the individual agents. Every game has 220 simulated days, and each day lasts 15 seconds of real time. The agents receive messages from the server on a daily basis informing the state of the game, such as the current inventory of components, and must send responses to the same server until the end of the day indicating their actions, such as requests for quotes to the suppliers. At the end of the game, the agent with the highest sum of money is declared the winner.

Normally, each manufacturer agent tackles separately important sub-problems of a supply chain: procurement of components, production and delivery of computers, and computer sales. Figure 1 summarizes the high level interactions between the various entities in the game.

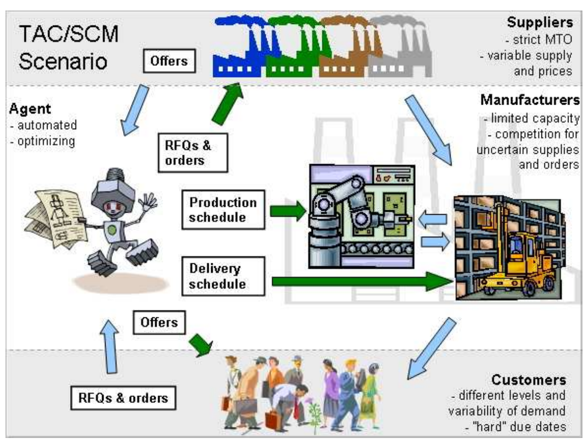


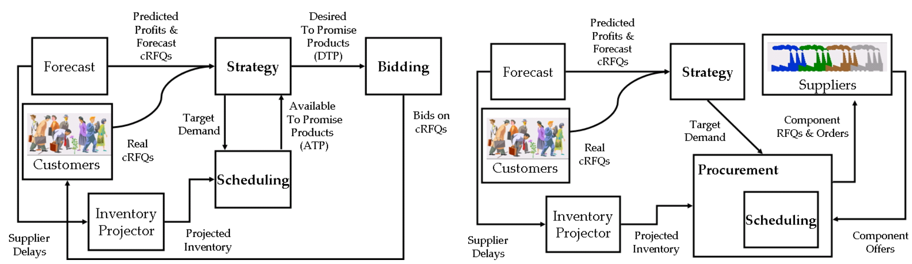

Fig. 1. Summary of the TAC SCM Scenario

### 2.1 Related Work

Development teams of TAC SCM agents have proposed several different approaches for tackling important sub-problems in dynamic supply chains. Deep Maize [Estelle et al. 2003] uses game theoretic analysis to factor out the strategic aspects of the environment, and to define an expected profitable zone of operation. SouthamptonSCM [He et al. 2006] presents a strategy for using fuzzy reasoning to compute bid prices on RFQs. TacTex [Pardoe and Stone 2004] presents machine learning techniques that were extended to form the customer bid price probability distributions in CMieux. The TacTex-05 team also offers considerable insight into the overall strategy behind their first-place agent in [Pardoe and Stone 2006]. The Botticelli team [Benisch et al. 2004] shows how the problems faced by TAC SCM agents can be modeled as mathematical programming problems, and offers heuristic algorithms for bidding on RFQs and scheduling orders.

## 3. CMIEUX

This section provides an overview of CMieux. For a more in-depth description of all modules the reader is directed to our technical report [Benisch et al. ].




(a) B2C Interaction Overview (b) B2B Interaction Overview

Fig. 2. Primary interactions between modules for B2C and B2B in CMieux.

### 3.1 Overview

Figure 2 shows the architecture of our CMieux supply chain trading agent, highlighting key interactions between its five main modules. The *bidding module* is responsible for responding to customer requests with price quotes. The *procurement module* sends RFQs to suppliers and decides which offers to accept. The *scheduling module* produces a tentative assembly schedule for several days based on available and incoming resources (i.e. capacity and components). The *strategy module* makes all high-level strategic decisions, such as what fraction of the assembly schedule should be promised to new customers and what part of the demand to focus on. The *forecasting module* is responsible for predicting the prices of components and the future demand.

Figure 3 gives a general overview of CMieux's main daily execution path. The agent begins by collecting any new information from the server, such as the new set of supplier offers, and customer requests. This information is fed to the forecast module, which updates its predictions of future demand and pricing trends accordingly. The forecast demand is given to the strategy module to determine what part of it our agent should target. From the set of forecast future RFQs the strategy module chooses a subset as the target demand. The procurement module then determines whether or not to accept each newly acquired supplier offer. All offers from suppliers are accepted unless they are too late to be useful, or too expensive to remain profitable. The scheduling module builds a tentative tardiness minimizing production schedule for up to twenty days in the future. The schedule includes the agent's actual orders, and the future orders composing the target demand. The target demand orders are used to determine how many finished PCs the agent has Available to Promise (ATP).

On the Business to Consumer (B2C) side, the strategy module uses the tentative ATP and the forecast selling conditions from the forecasting module to determine what the agent Desires to Promise (DTP). The DTP is used by the bidding module, along with learned probabilistic models of competitor pricing. The bidding module chooses prices to maximize the agent's expected profit, while offering the amount of products specified by the DTP in expectation.

The procurement module determines how many components are needed to reach the level of inventory specified by the strategy module. It compares the desired levels to the projected levels, and determines what additional components are needed. Each day the procurement module attempts to procure a fraction of the needed components based on the prices and availability predicted by the forecasting module.

---

(1) Update daily data structures with server information.
(2) **Forecast Module** → update forecasts.
 —Predict future orders and prices using regressions
 —Predict component arrivals based on observed delays
(3) **Strategy Module** → compute target demand.
(4) **Procurement Module** → accept supplier offers.
 —Accepts offers that are reasonably priced.
 —Accepts partial offers that are sufficiently large.
 —Accepts earliest offers that are not excessively late.
(5) **Scheduling Module** → make production schedule.
 —Uses dispatch scheduling and minimizes tardiness.
 —Available to Promise (ATP) products come from scheduled forecast orders.
(6) **Strategy Module** → compute target sales.

7a. **Bidding Module** → compute customer offers.
 —Probability models of competitor pricing are used to maximize expected profit and sell DTP in expectation.

7b. **Procurement Module** → send supplier requests.
 —Target demand is broken into requests to minimize expected offer cost.

---

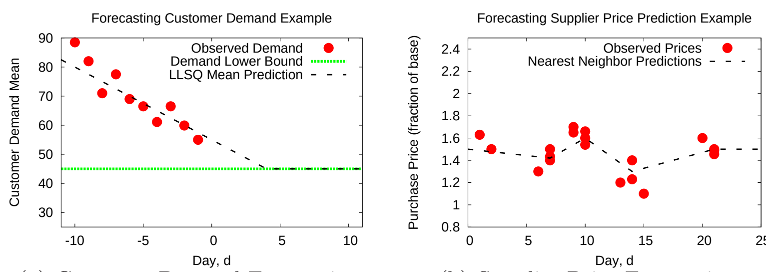
Fig. 3. Overview of CMieux's daily main loop.

### 3.2 Forecast Module

The forecast module is an important part of the pro-active planning strategies employed by CMieux. It helps inform a number of key decisions, such as the planning of RFQs sent to suppliers and the setting of target market shares for different end products, about the current and future market landscape.

The first responsibility of the forecast module is to predict a set of RFQs representative of those our agent expects to see in the future. The forecast module generates a representative set of RFQs by predicting the mean and trend of the customer demand from past observations. Each of the different product grades (high, medium and low) in TAC SCM is governed by its own mean and trend. The actual number of RFQs of each type received each day is drawn from a Poisson distribution with the mean of that type. The mean for each product type changes geometrically each day based on the trend (the trend is multiplied by the mean and the result is added to the subsequent day's mean), and the trend is changed by a small amount each day according to a random walk. The forecast module attempts to predict each of the changing mean and trend of the Poisson distribution governing demand separately, using a linear least squares fit of observations from the past several game days.

The second responsibility of the forecast module is predicting the selling price of each product, and the purchasing price of each component up to $D^{\mathsf{F}}$ days into the future. This information is useful to several of the other modules in the agent, such as the procurement module, that base decisions on current market conditions. The product selling prices are predicted in the same fashion as the demand trends. A linear least squares (LLSQ) fit is computed for the selling prices of each product over the past several game days (additionally, we enforce lower and upper bounds on the predictions to ensure they remain relatively conservative). The purchasing prices from a particular supplier are predicted using a nearest-neighbor (NN) technique based on historical prices quoted from that supplier. The forecast module predicts supplier prices on a particular day in the future by averaging observed quotes with nearby due-dates. Figure 4 shows examples of these two prediction techniques.




(a) Customer Demand Forecasting (b) Supplier Price Forecasting

Fig. 4. Examples of the techniques used by the forecast module to predict prices.

### 3.3 Strategy Module

The strategy module continuously re-evaluates and coordinates strategic decisions, including setting market share targets and selling quotas. These targets are continously tweaked to reflect both present and forecast market conditions.

More specifically, the strategy module determines what subset of the forecast customer RFQs the agent should aim to win (the "target demand") and what fraction of the its finished products the agent should plan on selling on any given day (the "desired to promise" products, or DTP). In other words, the strategy module modulates how the output of the forecast module impacts the procurement, scheduling and bidding modules (as illustrated in Figure 2).

On any particular day in the game, the strategy module must first determine the agent's target demand from the forecast demand. The goal of the strategy module is to target a fraction of the forecast demand that will lead to the highest overall profit (this is the agent's ultimate goal). The strategy module uses a heuristic to address this problem. When products are selling for a profit, it always targets exactly enough demand to stay at full factory utilization. The relative percentage of each product, or the *product mixture*, used to fill the target to full capacity is slightly adjusted each day based on the profit margin change of each product type. When the profit margin of a product increases (decreases), its relative percentage in the product mixture increases (decreases) slightly. When a product is no longer being sold for a profit, the strategy module calculates the product mixture in the same way. However, the mixture is post-processed so that the contribution of the unprofitable product is significantly decreased

After the target demand is computed by the strategy module, it is used by the scheduling module to develop a tentative production schedule for several days into the future (the *scheduling window*). The scheduling module uses information about incoming and available components, as well as previously committed orders. Using this information it determines when, if at all, each of the target orders will be produced (this process is described in Section 3.4). The part of this schedule assigned to filling target demand orders (as opposed to *actual* orders) indicates production that is not yet allocated to filling existing customer orders, or the available to promise (ATP) production.

The strategy module uses the ATP schedule to determine what the agent actually desires to promise each day (the DTP). In an effort to sell as little as possible and still maintain full factory utilization, the DTP consists of PCs appearing only in the first two day of the ATP schedule. The first two days of the ATP include un-promised finished products and un-promised products being produced that day.

### 3.4 Scheduling Module

The scheduling module continuously maintains a production schedule over a horizon of several days. This schedule reflects current contracts, forecast contracts and projected component inventory levels. It helps drive other planning decisions including which customer RFQs to bid on and which RFQs to send to suppliers. More specifically, the scheduling module makes a tentative production schedule for $D^{\mathsf{S}}$ days into the future. The inputs include a set of orders, $\hat{O}$, from the strategy module and the projected component inventory, $I$, for the remainder of the game. The orders in $\hat{O}$ represent the target demand of the agent and include both actual and forecast future orders.

The scheduling module uses a heuristic to sort orders according to "slack" (time before due date) and penalty, and a greedy dispatch technique to fill the production schedule. The dispatch technique proceeds as follows. It iterates through each day in the scheduling window and computes the priority of each unscheduled order during each iteration. The priorities are computed according to the Vepsalainen's apparent tardiness cost (ATC) dispatch rule [Vepsalainen and Morton 1987]. The ATC priority favors orders with large penalties and little time to complete, since these are likely to be orders that require the most immediate attention. The slack weighting parameter, $\alpha$, dictates the exact trade off in priority between slack and tardiness.

### 3.5 Bidding Module

The bidding module is responsible for responding to a subset of the current customer RFQs. Its goal is to sell the resources specified in the DTP at the highest prices possible.

The bidding module maintains a probability distribution, $G$, for each product type that specifies the likelihood of winning a particular RFQ for that product type at any price. The distributions, which constitute an aggregate opponent model, are learned off line using RFQs from previously played games. The distributions are trained on each historical RFQ based on its features (such as due date, penalty, and reserve price) and the features of the market at the time it was sent (such as the previous day's high and low winning bid prices). The bidding module uses the distributions to select offers that maximize expected revenue, subject to the restriction that the expected amount of products sold is less than or equal to the DTP.

The bidding module addresses the bidding problem separately for each product type, and reduces its task to a continuous knapsack problem (CKP) instance. The CKP is a variant of the knapsack problem classically studied in operations research. The CKP asks: given a knapsack with a weight limit and a set of weighted items – each with its value defined as a function of the fraction possessed – fill the knapsack with fractions of those items to maximize its value. In the CKP instance reduced from the bidding problem for a specific product type, the items are RFQs for that product with weights equal to their quantities. The weight limit in the CKP is the quantity of the product appearing in the DTP. The value of a fraction, $x$, of an RFQ, $r$, is the expected unit revenue that yields a winning probability of $x$. CMieux uses a binary search algorithm to solve the CKP instance for each product that is guaranteed to provide a solution within $\epsilon$ of optimal expected revenue. For full descriptions of the reduction to a CKP, the $\epsilon$-optimal algorithm, and the probability distributions used in CMieux the reader is directed to [Benisch et al. 2005].

### 3.6 Procurement Module

The procurement module handles all aspects of requesting and purchasing components. It is designed to rapidly adapt to changing market conditions. Each day, it considers sending requests with widely varying quantities and lead times in an effort to exploit gaps in current supplier contracts. By finding such gaps, or slow days for the suppliers, the agent ensures that its procurement prices tend to fall below its competitors. Each day, the procurement module performs two tasks: i.) it attempts to identify a particularly promising subset of current supplier offers, and ii.) it constructs a combination of RFQs to be sent to suppliers that balances the agent's component needs with identified gaps in current supplier contracts. The procurement module takes as input the set of recent supplier offers, the projected inventory, the target demand and the forecast pricing functions.

The module accepts supplier offers using a rule-based decision process. The agent begins by selecting offers that are satisfactory based on price, quantity and due date using historical data. In an effort to keep the agent's reputation with suppliers as high as possible[1], the agent first accepts all offers that satisfy the quantity and due date requirements of the corresponding RFQ ("full offers"). Next, if still needed, satisfactory offers with relatively large quantities ("partial offers"), or early due dates ("earliest complete offers") are also accepted.

[1] Maintaining a perfect reputation was identified as an important strategic goal for the 2005 competition.

Since offer prices, due dates and quantities are dictated by the specific requests they are offers for, the primary responsibility of the procurement module is requisitioning. The requisitioning procedure used in CMieux attempts to request some of the components it needs (that it has not already purchased) to maintain its target production levels, each day. Its main goal is to ensure that the prices offered in response to the requests are as low as possible. The requisitioning procedure chooses between many different lead times and quantities, based on the forecast supplier market landscape.

In order to determine what requests to send to suppliers, the procurement module computes, $\hat{I}$, the difference between the inventory required to maintain production levels specified by the target demand, and the projected inventory for the remainder of the game (i.e. the components that it needs but has not yet purchased). However, the components are not needed immediately, thus it can divide the purchasing of components in $\hat{I}$ across several days. To that end, the quantities specified in $\hat{I}$ beyond $D^{\mathsf{S}}$ days in the future (the scheduling window) are linearly depleted. This enables the agent to aggressively procure components within its scheduling window, so that late penalties are not incurred on existing contracts. In addition, it allows the agent to buy some of the components it needs well in advance, when they are likely to be cheapest.

The process of computing what specific requests to send to suppliers is then decomposed by component type (due to this decomposition the computation for each component type can be parallelized accross several CPUs). For each component type, the procurement module generates several sets of $K^{\mathsf{S}}$(the limit on RFQs sent each day) lead times and searches for the best set.

When considering which set of RFQs is best, the agent takes into account situations where it has all but one of the components required to assemble a particular type of PC (making it a *bottleneck component*). This situation can become more severe toward the end of the game as the agent faces the prospect of being stuck with mis-matched components. For example, our agent can have hundreds of motherboards, memory, and CPUs to make a specific product, and be missing only the hard drives. To address this issue, the procurement module artificially inflates the priority of RFQs that procure bottleneck components (such as the hard drives in the example)[2]. The inflation factor is increased as the agent nears the end of the game.

## 4. EMPIRICAL EVALUATION

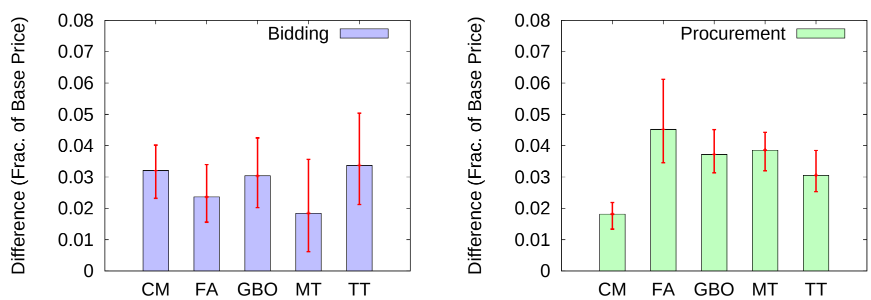
To validate the adaptive and dynamic techniques utilized in our agent we present the following set of empirical results taken from the 2005 TAC SCM seeding rounds[3]. We provide an in-depth analysis of our agent during what can be viewed as the most competitive phase of the competition, namely the 200 games played by the 25 agents participating in the second week of the seeding rounds. All agents at that stage have already been fine tuned over the course of about 600 games (two weeks of qualifying rounds, and one week of seeding).

Specifically, our results provide a statistical comparison between the performance

[2]This can be thought of as a coarse approximation of a component's marginal utility
[3]Competition data is available at `sics.se/tac/scmserver`

of the agents with the top 5 mean overall scores during the second week of the seeding rounds, namely CMieux (abbreviated CM), FreeAgent (FA), GoBlueOval (GBO), MinnieTAC (MT) and TacTex-05 (TT).

Performance was measured so as to identify those agents that were able to extract the highest sale price and lowest purchasing price in each game they played. Specifically, for each of the top 5 agents in each game it played in we computed how far it was from paying the least for its components and obtaining the most for its end products among the agents playing in that particular game. This was measured as the relative difference from the best average procurement price[4] and the best average selling price. For each of the top 5 agents we report the mean (with 95% confidence intervals) of these values across all of the games they each played in (see Figure 5).




(a) Bidding Analysis (b) Procurement Analysis

Fig. 5. The mean (with 95% confidence intervals) difference between each of the top 5 agents' average game unit price and the best unit price in the game, during the second week of the 2005 TAC SCM seeding rounds.

The bidding results for all 5 agents are relatively similar. As can be seen, each of the top 5 agents is on average within about 3% of the base price from being the best in its games. However, while MinnieTAC (MT) was the closest to the best agent in its games, with an average difference of about 2% of the base price, there is no statistically significant difference between any of the top 5 agents (as evidenced by their overlapping confidence intervals). On the other hand, the procurement results show that our agent, CMieux (CM), is significantly closer to being the best than all 4 of the other top 5 agents. These results seem to validate CMieux's approach to tightly coordinating its bidding, planning and procurement operations. They also suggest that the agent's approach to optimizing the RFQs it sends to suppliers (*requisition process*) was significantly more effective than the procurement strategies implemented by its competitors.

## 5. CONCLUSIONS

This paper presented a high level view of the interactions between the different modules composing CMieux, Carnegie Mellon University's 2005 TAC SCM entry,

[4]All prices are considered as fractions of the corresponding product or component's base price.

as well as brief descriptions of its decision making processes. CMieux's architecture departs markedly from traditional Enterprise Resource Planning architectures and commercially-available supply chain management solutions through its emphasis on tight coordination between supply chain bidding, procurement and planning.

CMieux finished 4th in the 2005 seeding rounds of the TAC SCM tournament and reached the competition's semifinals. In this paper, we presented a more in-depth analysis of the agent's performance based on 200 games involving agents entered by 25 different teams during what can be seen as the most competitive phase of the 2005 tournament. The results show that our agent performed on par with the best in its bidding while significantly outperforming these agents in terms of procurement. These results seem to validate CMieux's approach to tightly coordinating its bidding, planning and procurement operations. They also suggest that the agent's approach to optimizing the RFQs it sends to suppliers (*requisition process*) was significantly more effective than the procurement strategies implemented by its competitors.

## 6. ACKNOWLEDGMENTS

The research reported in this paper has been funded by the National Science Foundation under ITR Grant 0205435.

### REFERENCES

Arunachalam, R. and Sadeh, N. 2005. The supply chain trading agent competition. *Electronic Commerce Research Applications 4,* 1.

Benisch, M., Andrews, J., and Sadeh, N. 2005. Pricing for customers with probabilistic valuations as a continuous knapsack problem. Tech. Rep. CMU-ISRI-05-137, School of Computer Science, Carnegie Mellon University. December.

Benisch, M., Greenwald, A., Grypari, I., Lederman, R., Naroditsky, V., and Tschantz, M. 2004. Botticelli: A supply chain management agent. In *Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*.

Benisch, M., Sardinha, A., Andrews, J., and Sadeh, N. Cmieux 2005: Design and analysis of carnegie mellon universitys entry in the supply chain trading agent competition. Tech. rep.

Chopra, S. and Meindl, P. 2004. *Supply Chain Management.* Pearson Prentice Hall, New Jersey.

Collins, J., Arunachalam, R., Sadeh, N., Eriksson, J., Finne, N., and Janson, S. 2005. The supply chain management game for 2005 trading agent competition.

Estelle, J., Vorobeychik, Y., Singh, M. P. W. S., Kiekintveld, C., and Soni, V. 2003. Strategic interactions in a supply chain game.

He, M., Rogers, A., Luo, X., and Jennings, N. R. 2006. Designing a successful trading agent for supply chain management. In *Proceedings of AAMAS'06*.

Pardoe, D. and Stone, P. 2004. Agent-based supply chain management: Bidding for customer orders. In *AAMAS-04 Workshop on Agent-Mediated Electronic Commerce*.

Pardoe, D. and Stone, P. 2006. Predictive planning for supply chain management. In *Proceedings of Automated Planning and Scheduling'06*.

Sadeh, N., Hildum, D., Kjenstad, D., and Tseng, A. 1999. Mascot: an agent-based architecture for coordinated mixed-initiative supply chain planning and scheduling. In *Proceedings of Workshop on Agent-Based Decision Support in Managing the Internet-Enabled Supply-Chain at Agents'99*.

Vepsalainen, A. and Morton, T. 1987. Priority rules for job shops with weighted tardiness costs. *Management Science 33,* 8, 1035–1047.