The TAC SCM Prediction Challenge

David Pardoe
Department of Computer Sciences
The University of Texas at Austin
dpardoe@cs.utexas.edu

Last update: June 18, 2007

1. Overview

In order to effectively manage a supply chain, a TAC SCM agent must be capable of performing a number of interrelated tasks. In the TAC SCM Prediction Challenge, agents will be evaluated on their ability to perform a single one of these tasks in isolation: making predictions about prices. In particular, agents will make daily predictions over the course of a number of TAC SCM games about four different types of prices: current and future computer prices, and current and future component prices.

Instead of making predictions about live TAC SCM games in which they are participating, agents will make predictions on behalf of another agent named SCMPredictionAgent (referred to as PAgent hereafter). Before the competition, the organizers of the challenge will run a number of games using PAgent and other agents.[1] The identities of these other agents and the resulting game logs will not be made available to participants. During the competition, these games will be "re-played" from the game logs. For each day of each game, participating agents will receive the exact messages sent to PAgent (incoming messages), as well as the messages it sent to the game server in response (outgoing messages). As this is exactly the information that would be available to an agent during a live game, it is hoped that existing TAC agents can be easily modified to make use of messages presented in this fashion. In addition to these incoming and outgoing messages, each agent will also be given a set of predictions that must be made before the information for the following day will be sent. The predictions consist of:

1. The price at which each RFQ sent from customers on the current day will be ordered (i.e., the lowest price that will be offered by a manufacturer),
2. The median price at which each of the 16 types of computer will sell in $N_{computer}$ days,
3. The price that will be offered for each RFQ sent by the PAgent to a supplier on the current day, and
4. The price that will be offered for each of a number of provided RFQs that will be sent by the PAgent to suppliers in $N_{component}$ days.

---

[1] All games will conform to the latest TAC SCM specification, available at http://www.sics.se/tac/tac07scmspec.pdf

There are a number of benefits to running the competition using logs from completed games instead of using live games. First, there is no restriction on the number of agents that may compete head to head at one time. Second, each agent will receive exactly the same information about the state of each game and will be asked to make the same predictions. Finally, in live games there would be an incentive for agents to behave differently than in normal TAC SCM games. For instance, agents might use their available RFQs for probing supplier prices rather than requesting needed components, or they might attempt to manipulate prices in order to make past predictions accurate.

Agent performance will be evaluated separately for each of the four prediction categories. Root mean squared error will be used as the scoring metric, and all errors will be measured as a fraction of the base price. Agents will be ranked in each category, and the overall winner will be the agent with the highest average rank over all four categories. Average RMS error over all categories will be used as a tiebreaker. Agents are not required to make predictions in all categories – participants interested only in evaluating their performance in certain categories may direct their agents to omit the predictions for the other categories.

2. Competition Format

To test the ability of agents to make predictions for games with various competitors, agents will be required to make predictions for C sets of games. In each set, the PAgent will have competed against a different group of five competitors. Each set will contain G games, meaning that agents will have a chance to improve their predictions through repeated experience with the same group of competitors. Predictions will be made for one game at a time, and agents will be informed when one set of games has ended and a new set is beginning.

There are two versions of the code: a standalone version and a client/server version. The standalone version is useful for agent design and requires all game logs to be available locally. In the client/server version that will be used for the competition, agents have no direct access to the logs and must connect to a remote server. The competition will take place on a single day during the TAC SCM finals. Because there is no need for all agents to make their predictions simultaneously, the fixed game times and time limits for each game day of the standard TAC SCM game will not apply. A start time and end time, spanning H hours, will be announced. Agents will be free to contact the server to begin at any time after the start time, and they may make predictions at any pace so long as they finish before the end time.

3. Communication

An agent receives two types of communication. The first is a notification that a new game has started. Along with this notification, the agent is told whether the new game involves a new set of competitors, or the same competitors from the previous game. The second is a notification that a new day in the current game has started. Along with this notification, the agent receives a list of the messages received by and sent by the PAgent (in the exact same format as in the original game), and a data structure called a *prediction object* that is used to make predictions.

The prediction object specifies what predictions need to be made and also stores the predictions made by the agent. Details for each of the four prediction categories are given below:

1. Predictions of prices for each of the current day's RFQs from customers are required on all but the first day and the last two days. The prediction object indicates whether predictions are required, and for each RFQ, an agent makes a prediction by specifying the RFQ ID and the predicted unit price. If the RFQ does not result in an order, the prediction will be ignored when accuracy is evaluated. Therefore, agents do not need to be concerned with *whether* an order will be result, only what the price will be if there is an order.
2. Predictions of median prices in $N_{computer}$ days for each of the 16 computer types are required on all but the last $N_{computer}$ + 2 days. The prediction object indicates whether predictions are required, and for each type of computer, an agent makes a prediction by specifying the product ID and the predicted median unit price. If no computers of a certain type are sold in $N_{computer}$ days, the prediction for that type will be ignored when accuracy is evaluated.
3. A prediction of the price that will be offered by the supplier is required for each RFQ sent by the PAgent to a supplier. The PAgent does not send any RFQs to suppliers during the last 10 days of each game. An agent makes a prediction by specifying the RFQ ID and the predicted unit price. If an RFQ results in no offer (due to the reserve price) or an offer (or offers) with modified quantity or due date, the prediction for that RFQ will be ignored when accuracy is evaluated.
4. For each of the 16 pairs of a supplier and a component that it supplies, the prediction object provides a zero-quantity RFQ that will be sent by the PAgent in $N_{component}$ days with a due date chosen at random between D_{min} and D_{max} (or the number of days remaining, if less than D_{max}) days after the date the RFQ is sent. Because the PAgent sends no RFQs during the last 10 days of a game, predictions for future RFQs do not need to be made during the last $N_{component}$ + 10 days of the game, and the list of these future RFQs provided by the prediction object will be empty. For each RFQ, an agent makes a prediction by specifying the RFQ ID and the predicted unit price.

4. Parameter Values

This is a table of game parameters.

Parameter Description	Symbol	Value
Number of days into the future for computer price predictions	$N_{computer}$	20 days
Number of days into the future for component price predictions	$N_{component}$	20 days
Minimum of the date range for the RFQs used as future RFQs to predict	D_{min}	5
Maximum of the date range for the RFQs used as future RFQs to predict	D_{max}	30
Number of different sets of competitors for which games will be played	C	3 sets
Number of games in each set of games	G	16 games
Time available for completing all predictions	H	8 hours

Appendix: The SCMPredictionAgent

Although predictions could be made on behalf of any agent from a completed game, the use of a single agent (PAgent) for which source code is available simplifies the task of participants by helping them to understand exactly what behavior to expect from the agent. For example, understanding how the agent decides to send RFQs to suppliers may be useful when designing a method of making predictions about supplier prices. PAgent was designed to be as simple as possible and to behave in a consistent and predictable manner while still exhibiting somewhat reasonable behavior. This section provides a basic overview of the aspects of PAgent that are most relevant to the competition.

PAgent considers RFQs from customers in a random order. For each RFQ, PAgent determines a random offer price. If the price is below the reserve price, PAgent attempts to reserve the resources that will be needed to produce the requested computers. If sufficient resources are available, then PAgent makes the offer.

For each RFQ, the random offer price is chosen as follows. Let l and h be the lowest and highest reported prices, respectively, from the previous day for the requested computer type, and let $range = 1.5 * \max(h - l, 50)$. Let x be chosen uniformly randomly from [0,1], and let $p = 1\text{-}x^2$. The price chosen is then $l + p * range$. Prices chosen in this way fall within a range where it is reasonable to believe that the order might be won, but the prices are skewed toward the high end of this range where winning is less likely. As a result, PAgent generally makes a large number of offers, and a moderate fraction of the offers usually result in orders. To further reduce (but not necessarily eliminate) the chances of receiving unprofitable orders, PAgent increases p by 0.5 before determining the price to offer when h is below PAgent's estimated cost to produce the requested computer type.

PAgent sends a fixed set of RFQs to suppliers on the first day of each game (500 of each component from each supplier due on days 5, 10, 15, 20, and 25), sends no RFQs during the last 10 days of each game, and follows a fixed procedure on all other days. All supplier offers are accepted, so PAgent's reputation will always be perfect.

The procedure used on most days is to send two zero-quantity RFQs for each supplier-component pair, and to use the remaining three RFQs to request needed components. The first zero-quantity RFQ has a due date chosen randomly between D_{min} and D_{max} (or the number of days remaining, if less than D_{max}) days in the future. (This is the RFQ that will require a prediction $N_{component}$ days in advance.) The second zero-quantity RFQ has a due date chosen randomly between $D_{max} + 1$ and 60 days in the future, and is sent only if the due date is before the end of the game.

For each component type, PAgent generates its remaining RFQs as follows. Three dates are chosen as due dates for RFQs: one between 5 and 10 days, one between 11 and 20 days, and one between 21 and 30 days in the future. For a due date d days in the future, PAgent determines n, the number of additional components that would be needed to keep its projected inventory above 800 on that date, assuming that the average rate of component use over the past 10 days continues. The quantity of components to be requested is set to $n * (1 - 0.03*(d\text{-}5))$. By reducing the fraction of the needed components that will be requested in this way, PAgent avoids relying primarily on long-term orders. The resulting quantity is then requested from the supplier, or split between requests to each supplier if there are two suppliers for the component. The reserve price is set to the component base price. For due dates beyond the end of the game, no RFQs are sent.