Visualization and Analysis Methods for Comparing Agent Behavior in TAC SCM

(Extended Abstract)

William Groves, John Collins, Maria Gini*
Department of Computer Science and Engineering, University of Minnesota
{groves, jcollins, gini}@cs.umn.edu

ABSTRACT

We are targeting for analysis a complex, multi-agent, repeated market simulation in which agents compete to maximize overall profit over a finite time horizon. The Trading Agent Competition for Supply Chain Management (TAC SCM), which has been run annually as a tournament for five years, is designed to model realistic supply chain and market mechanisms. In this paper, we introduce our extensible analysis framework that can be used for posing and answering new questions related to understanding market mechanisms and agent behavior.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

General Terms

Design, Experimentation

1. OVERVIEW AND MOTIVATION

Graphical visualization of complex data allows humans to efficiently comprehend observations in a complex system. TAC SCM possesses considerable complexity which is a significant barrier to quickly understanding the emergent behavior of the competing agents. Each run of the game consists of six autonomous agents competing for 220 business “days” to buy computer components, build computers, and sell computers to customers. The agents act as consumers in one market and sellers in the second.

In this paper, we illustrate an analysis method that can be used to compare simulation runs as well as to highlight differences in behavior between the competing agents. First, we show how game data can be parsed as XML and read into a database. Then, we show how to pose analysis questions to the database in SQL to obtain useful visualizations. We show how visualization can provide insight to the analysis question posed. The analyses enabled by these visualization tools have been invaluable in directing development efforts for our trading agent entry: MinneTAC.

*Partial support provided by NSF under grant IIS-0414466

Cite as: Visualization and Analysis Methods for Comparing Agent Behavior in TAC SCM, (Extended Abstract), William Groves, John Collins, Maria Gini, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1367–1368

An analysis question we have used for our own agent development will be provided as an example of how to construct an analysis module in our framework. The publicly available tools for analysis of TAC SCM games [1] are insufficient for directing many aspects of agent design. We seek to lower the barrier of entry for participation in TAC SCM by making our analysis tools freely available to the community.[1]

2. METHODOLOGY

We have developed visualizations that can facilitate several goals including measuring agent performance in various aspects of the game, verification of mechanism design, comparisons between agents, and comparison of an individual agent across games. Previous work has shown that basic game statistics such as ranking, overall profit, or utilization are extremely unreliable comparative measures because of the large variability between games [2]. Through targeted analysis similar to the analysis shown later, we can quickly detect specific performance improvements in our agent.

Database Storage of Game Messages

To begin any game analysis, a corpus of previous game data must be identified. A complete set of all simulation messages is available from the competition server at the conclusion of each game. From this publicly available data, the actions of each agent can be analyzed. The log analysis tool provided with the game server allows game messages and internal server state information to be exported. For efficient querying and analysis, we export the game data in its native XML format and then import the data into an SQL database. The XML code below provides an actual message sent from the server to an agent:

```
<message sender="15" receiver="8" time="1115456844785">
<rfqBundle>
<rfq id="19" product="400" quantity="300" duedate="3"
penalty="0" reservePricePerUnit="320" />...
</rfqBundle></message>
```

Each game message is represented as a row in the database. A parser converts each message from XML into its equivalent SQL. The message above generates the following SQL:

```
INSERT INTO rfqs (simid,timeunit,sender,receiver,id,
product,quantity,duedate,penalty,reservePrice)
VALUES ("tac3.sics.se:3719",0,15,8,19,400,300,3,0,320)
```

Each game has over 500,000 messages requiring uncompressed storage of around 100MB. Database systems are best equipped to handle non-sequential querying for data sets of

[1]Information available at: `www.cs.umn.edu/~groves/tac/`

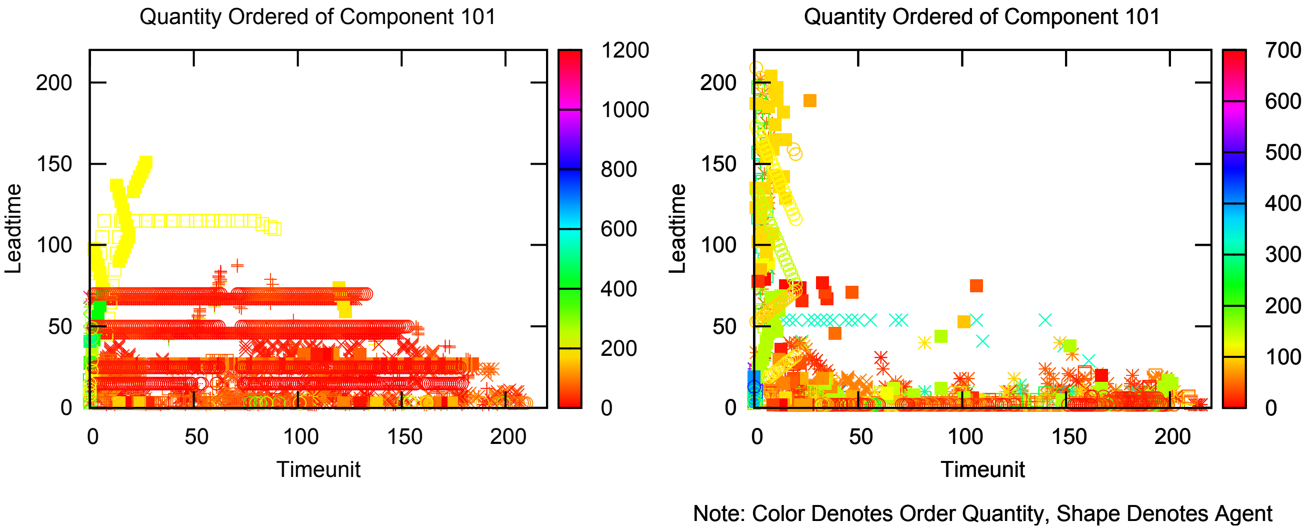

Figure 1: A plot of ordering behavior in two tournament games: (a) 2005-#3719, (b) 2008-#786.

this magnitude. Once the game database is populated with game messages, it is possible to run SQL queries against it.

Analysis Questions as SQL Queries

The analysis framework described here will be demonstrated with an example taken from our agent development efforts: we will consider the use of long lead time ordering behavior in the top ranking TAC agents. It is common practice for agents in the game (as well as companies in the real-world) to use long-term contracts to "lock-in" fixed prices for future supply chain needs. We seek to characterize the current use of long-term ordering in the supply chain game and examine how its use has changed as the competition has evolved over time. While an agent can "lock-in" relatively low prices by ordering far in advance, there are two penalties to this approach: interest on the down payment is incurred, and the agent may be less competitive later in the game because it cannot easily adapt its production strategy to accommodate changes in market conditions. In the worst case, an agent may buy a large number of parts far in advance which are unsalable due to a slump in demand; this will reduce the agent's overall profit for the game. Striking a balance between low procurement prices and production flexibility is important, so we will examine how top agents have used long-term procurement by looking at competition games.

We would like to ascertain how prevalent long lead time ordering is among top agents in the competition. This analysis task involves writing a series of SQL queries to obtain statistics on the quantity of components ordered for each unique combination of component, lead time, and due date. One possible approach to answering this question involves aggregating data points from the RFQ (request for quote) table in the populated database of game messages. The SQL query for this particular analysis question is shown below:

```
SELECT sender, product, timeunit, leadtime,
sum(quantity) FROM rfqs WHERE quantity > 0
GROUP BY product, sender, timeunit, leadtime
```

The rows resulting from the query are then separated by product, and data for each product can be analysed separately. The next section addresses visualization of the query.

Building Graphs from Data

While some analysis questions can be answered with a single number or a small matrix of data, many questions are best answered through a visual inspection of large amounts of data as a graph. The visualizations in Figure 1 are developed using the popular plotting tool Gnuplot. The long lead time query is run against game data from both a 2005 and a 2008 TAC SCM finals game. The behavior in the 2008 game shows a strong preference toward a significant quantity of very long-term ordering (along the y-axis) supplemented by short-term orders (along the x-axis). The 2005 figure shows that only one agent engages in significant long-term ordering behavior, while the rest spread their offers at shorter, fixed lead times. From these graphs, we see that fixed lead time procurement strategies have fallen out of fashion in favor of randomized long-term strategies supplemented by short-term ordering. Such analysis questions can be efficiently posed and answered using our analysis framework.

3. DISCUSSION

The methodology outlined here demonstrates a database mechanism for storing game data for complex analysis. This enables efficient filtering to isolate and examine specific agent or market behavior. Posing analysis questions as SQL queries streamlines the analysis process and reduces the need for ad-hoc analyses based on aggregate data. The ability to quickly flesh-out and test new analysis questions not covered by existing tools facilitates making progress in the domain.

4. REFERENCES

[1] M. Benisch, J. Andrews, D. Bangerter, T. Kirchner, B. Tsai, and N. Sadeh. Cmieux supply chain trading analysis and instrumentation toolkit. Technical Report CMU-ISRI-05-127, Carnegie Mellon University, 2005.

[2] E. Sodomka, J. Collins, and M. L. Gini. Efficient statistical methods for evaluating trading agent performance. In *AAAI*, pages 770–775. AAAI Press, 2007.