CMSC 691A Software Agent Project Report

Youyong Zou
Lianhe Shao

TAC
ICMAS-00
Trading Agent Competition

1 . What is TAC

The Fourth International Conference on Multiagent Systems (ICMAS-00),to be held in Boston in July 2000, will feature a Trading Agent Competition (TAC), in which software agents developed by research groups, students, and others from all over the world will compete in a challenging market game.

This game will be hold on the Michigan Internet AuctionBot platform, a configurable auction server that implements a diverse set of auction rules and thus enables a wide variety of market games. The AuctionBot supports participation by software agents through a TCP-based agent programming interface (API).

In the TAC shopping game, each "shopper" (i.e., an entrant to the competition) is a travel agent, with the goal of assembling a number of travel packages (from TACtown to Boston, for a week in July 2000). Each agent is acting on behalf of a number (set at 8) of customers, who express their preferences for various aspects of the trip. Travel packages consist of the following: A round-trip flight, A hotel reservation, and entertainment Tickets .All three types of goods will be traded in separate markets with different rules.The objective of the travel agent is to maximize the combined preference of its customers. One round of game will last 15 minute.

2.Design

One TAC customer package include:

** A round-trip flight*

** A hotel reservation for every night between arrive and departure, it may be good hotel or bad hotel.*

**Tickets to some of the following entertainment events*

----- Boston Red Sox Baseball Game

----- Boston Symphony

-----Boston Theater production of "The Phantom of the Opera

If the customer get full package, he will be satified, and trading agent will get a score:

$$u = 1000 - travel\_penalty + hotel\_bonus + fun\_bonus$$

where

travel_penalty = 100(|AP - AD| + |DP - DD|)*
*hotel_bonus = GH? * HU*
*fun_bonus = BT? * BRS + ST? * SY + PT? * PH*
A single preferred arrival date (AP).
A single preferred departure date (DP).
A reservation value for upgrading to the better hotel (HU).
A reservation value for each type of entertainment event (BRS, SY, PH).
An actual arrival date (AD).
An actual departure date (DD).
Good hotel indicator (GH? in {0,1}).
A ticket indicator for each event type (BT?, ST?, PT?, each in {0,1}).

2.1 Find best plan

when we organize a travel package for a customer, we will change arrive date from 1 to 4, change departure date from 2 to 5, change hotel between good hotel and bad hotel. Compute a u value for every solutions and find the best plan, which maxize the u value. If the u value is bigger than 0, that means we can make money from this customer, we will bid for this package.

2.2 Change plan

When the auction is going on, the price of air ticket and hotel is changing. If we find there is a better plan, we can still change to this better plan. The prerequirement is that we have not win anything we currently bidding. (we donot want to waste money for one customer) and this better solution's auction is still open.

2.3 Air ticket

The price of air ticket is keeping changing and it is continuous clear. The number of air ticket is unlimited. We will set two reserve time. First reserve time is 500 second, Second reserve time is 100 second. Before first reserve, we will bid a very low price and try our lucky. Between those two reserve time, we will bid price a little less than auction's askprice. After second reserve time, there is no too much time left, we will bid askprice+1 and make sure we can get the ticket.

2.4 Hotel

After compute the u value, we will bid the hotel we need. Because hotel bid cannot withdraw, so only if the hotels we are bidding are failed, we will not change hotel plan. If the hotel bidding is failed, we will change to new plan which auction is still open.

2.5 Entertainment ticket.

That is the most difficulty part of the bidding agent.
We define four matrix :

Availavle[custno, day]: if the customer is intown in this day, it is 1 ,else 0;
Endow[TktType,day]: how many ticket I have for every day, each type;
Prefer[custno, TkeType]: how much this customer would like to pay for this ticket

3. Implement

API and BiddingAgent software

AuctionBot provide two tools for designing agents.

- *The lowest-level AuctionBot API software implements functions to send and receive messages via TCP with the AuctionBot server. The API software is available in C++, Java, and Mathematica.*
- *The bidding agent software provides higher level support for managing bids and information about auctions. The bidding agent software is available in C++ with STL for Unix. The AuctionBot bidding agent classes serve as a middle level interfaces between an agent bidding in auctions and the Michigan Internet AuctionBot itself. The bidding agent classes provide a layer above the AuctionBot API to help synchronize information between the AuctionBot and the agent. It also provides some sample trading agents,*

Since C++ get more support in AuctionBot, This project will use C++ as programming language.
Click here to Download software.

We need to design an agent class :UmbcTAC1agent , which is a subclass of TradAgentShell, TradAgentShell is a subclass of Bidder.

Bidder classes automatically changes bids (according to the bidding policy programmed by the programmer/user) and keeps track of information from the AuctionBot, including: price quotes, transactions, clearing prices, auction closings, bid expirations, bid acceptances, and bid rejections.

The TradingAgentShell is a skeleton agent shell you can use to derive agents. It is derived from the Bidder class and contains objects for holding bids in the various auctions and a TACData object for TAC specific data assigned to an agent. The data structure for holding the bids for the various auctions is mBids, which is basically a map from the auctionId to the agent's bid object for that auction. This shell implements no strategy at all and you need to plug-in the desired strategy in the function UpdateBids().

Main Methods of UmbcTAC1agent :

- *Constructor(): calles constructors for parent class and initalize workData;*
- *Destructor(): Withdraws all bids placed by this agent, disconnects from the AuctionBot.*
- *UpdateBids(): Constructs bids for all auctions using a simple strategy and submits the bids*
- *Terminate(): Use TradAgentShell Terminate check logic.*
- *ConstructAllBids(): Goes through all client requirements and generates bid strings for all the AgentPointBidSchedules.*

We will use example agent TradeAgent3 as skeleton. TradeAgent3 use some simple auction algorithms . We need to have more complex auction algorithms for out trade agent. There are three kinds of goods in trade game, each goods will be auctioned by different auction type.

- *Flights: All buy bids are accepted, matching is based on earliest submission time, quotes and clears are continuous, there is one seller, multi-point buy bids are allowed, and only the ask price is quoted.*
- *Hotels: standard English ascending, except for fixed final clear. (Ascending, Mth price, multiple buy points in bid allowed, no withdrawal, continuous quotes, quote is ask price, must beat quote, no termediate clears)*
- *Entertainment Tickets: Continuous double auction, any type of bid allowed.*

So, we need design different auction algorithms for them. The algorithms should be simple and efficient. The auction algorithms will be added in methods UpdateBids() and ConstructAllBids().
There are some auction algorithms in reference page.

4.Test

Communication between agents and auction server is TCP based socket. The auction server is auction-sun.eecs.umich.edu, port: 6500
We can test the agent software in the TAC practice game page

5. Future Work

I have submited letter of participate to ICMAS-00 TAC. So I will keep improving my bidding agent and try to win the Game.
Timetable is:

- June 17, 2000 Preliminary rounds
- July 7, 2000 TAC Contest, Boston MA.

I need to improve:

- Interexchange of hotel ticket.
- Better allocation algorithm for Entertainment ticket.
-

6.Reference

1.Flexible double auctions for electronic commerce: Theory and implementation (PR Wurman, WE Walsh, and MP Wellman).Decision Support Systems 4:17-27 1998
2. Real time issues for Internet auctions (MP Wellman and PR Wurman). First IEEE Workshop on Dependable and Real-Time E-Commerce Systems, June 1998.
3. The Michigan Internet AuctionBot: A configurable auction server for human and software agents (PR Wurman, MP Wellman, and WE Walsh). Second International

Conference on Autonomous Agents, May 1998.
4. The Auction Manager: Market middleware for large-scale electronic commerce (T Mullen and MP Wellman). Third USENIX Workshop on Electronic Commerce, September 1998.
5. Introduction to the special section on Knowledge-based construction of probabilistic and decision models (JS Breese, RP Goldman, and MP Wellman). IEEE Transactions on Systems, Man, and Cybernetics 24(11):1577-1579, 1994.
6. A Parametrization of the Auction Design Space, (PR Wurman, MP Wellman, WE Walsh). Games and Economic Behavior
7. Market-aware agents for a multiagent world (MP Wellman and PR Wurman). Robotics and Autonomous Systems 24:(3-4) (1998) 115-125.
8. Control Architecture for a Flexible Internet Auction Server (PR Wurman, MP Wellman, WE Walsh, and KA O'Malley). First IAC Workshop on Internet Based Negotiation Technologies, Yorktown Heights, NY, March, 1999.
9. Equilibrium Prices in Bundle Auctions. (PR Wurman and MP Wellman), Santa Fe Institute Working Paper 99-09-064.
10.The DiMalloc Project : Efficient Market Protocols and Resource Allocation Algorithms
11. http://www.enersearch.se/~ygge/ : Dr.Fredrik Ygge 's website, include some article about auction algorithms, and even source codes.
12. http://www.econ.umn.edu/~czheng/f96_5107h/games.htm collect of Game Theory, include article and source.
13. R. Guttman, A. Moukas, and P. Maes. "Agent-mediated Electronic Commerce: A Survey." To appear, Knowledge Engineering Review, Vol. 13:3, June 1998.
14. R. Guttman and P. Maes. "Cooperative vs. Competitive Multi-Agent Negotiations in Retail Electronic Commerce." Proceedings of the Second International Workshop on Cooperative Information Agents (CIA'98), Paris, France, July 3-8, 1998.