CM30174: E-commerce and Agents: Assessed CourseWork 1: Implementing a Trading Agent

Dr. Julian Padget and Dr. M. De Vos
Dept of Computer Science
University of Bath
{jap, mdv}@cs.bath.ac.uk
Unit home page: `http://www.cs.bath.ac.uk/~jap/CM30174`

1 Introduction

The goal of this project is to design and implement the decision-making aspects of an agent able to participate in the International Trading Agent Competition (TAC)[1]. The work is to be carried out in pairs.

2 Objectives

At the end of this coursework you will be able to

- develop a very simple agent-based system.
- assess different strategies for implementing decision-making. behaviour in the context of this project.
- build on existing code.

3 @Lic and the International Context

The @LIS Technology Net Project will create a highly innovative teaching and experimentation environment spanning Europe and Latin America. The environment will operate as an always-on network — connecting all partners and populated with autonomous software components able to interact dynamically with one another and provide services to their users.

The @LIS Technology Net Project will build on the global Agentcities testbed to use the latest Agent, Web / Internet technologies. Important results will include:

[1]http://www.sics.se/tac/

- Agentcities Agent and Web/Internet technology infrastructure linking Chile, Costa Rica, Cuba, Mexico, Spain, Italy and the UK as well as many other parts of the world — enabling software systems deployed by any partner to communicate with those deployed by others.
- A virtual teaching environment enabling students, learning professionals and researchers to gain hands-on experience of using cutting-edge Agent, Web, and Internet technologies to create complex dynamic on-line applications.
- An application demonstrator providing personalised cultural/tourism services accessible from both wireline (PCs) and wireless (mobile, roaming PDAs or other handheld devices) constructed from dynamically composed services deployed across the partner countries.

4 Project

The following subsections provide the project description, functional and non-functional requirements, the guidelines for installing the necessary code to start your project and the requirements for the report to be submitted for assessment.

4.1 Background Information

The International Trading Agent Competition (TAC) is a research oriented competition that allows universities and research centres to develop and experiment with behaviour of agents participating in trading. TAC uses a fixed game to test the agents that take part. In a TAC trading game, there are 8 software agents (entrants to the competition) that compete against each other in a variety of auctions to assemble travel packages for their individual customers preferences for the trip. A valid travel package for an individual customer consists of (i) a round trip flight during a 5-day period (between TAC-town and Tampa) and (ii) a stay at the same hotel for every night between their arrival and departure dates. Moreover, arranging appropriate entertainment events during their the trip increases the utility for the customers. The objective of each agent is to maximise the total satisfaction of its 8 customers (i.e. the sum of customers' utilities). Customers have individual preferences with regards to their travel dates, the hotel and the entertainment they want to attend. This data is randomly generated by the TAC-server in each game.

Each agent communicates with the TAC-server through a TCP-based agent programming interface (API) in order to obtain current market information and to place its bids. An individual game lasts 12 minutes and involves 28 auctions. The auction for each good type has different rules:

- *Flights*. TACAIR is the only airline selling flights. Tickets for these flights are unlimited and are sold in single seller auctions. There are 8 such auctions (4 for outward flights on days 1 to 4 and 4 for homeward flights on days 2 to 5). Flight ask prices update randomly, every 24 to 32 seconds, by a value drawn from a range determined by the elapsed auction time and a randomly drawn value. Flight auctions clear continuously during the game. Thus, any buy bid an agent makes that is not less than the current ask price will match immediately at the ask price. Those bids not matching immediately remain in the auction as standing bids. In most cases, the earlier the flight is booked, the cheaper it is.

- *Hotels*. There are two hotels in Tampa: Tampa Towers (TT) and Shoreline Shanties (SS). TT is nicer than SS. Hotel rooms are traded in 16th price (that is the winner pays the 16th highest bid) multi-unit English Auctions. Overall, there are 8 hotel auctions (for each combination of hotel and night apart from the last one), which close randomly one by one at the end of every minute after minute 4 of the game. A hotel auction starts at \$0 and clears and matches bids when it closes, i.e. 16 rooms are sold at the 16th highest price and this price is immediately updated as new bids come in.
- *Entertainment*. Each agents is randomly endowed with 12 entertainment tickets at the beginning of the game, choseon the following basis:
  - One bundle of four of a particular type on day 1 or day 4.
  - One bundle of four of a particular type on day 2 or day 3.
  - One bundle of two of a particular type (different from above) on day 1 or day 4.
  - One bundle of two of a particular type (different from above) on day 2 or day 3.

  All agents can trade their tickets in a continuous double auction (CDA). Overall, there 12 CDAs (for each kind of entertainment for each of the days). Bids match at the price of the standing bid in the CDA. An entertainment package is feasible if none of the tickets are for events on the same day and all tickets coincide with the nights the customer is in town. No additional utility is obtained for a customer attending the same type of entertainment more than once.

A customer's utility from a valid travel and entertainment package is given by:

$$\text{Utility} = 1000 - \text{TravelPenalty} + \text{HotelBonus} + \text{FunBonus}$$

where TravelPenalty = $100*(|AD-PAD|+(|DD-PDD|))$ (AD and DD are the customer's actual arrival and departure dates). HotelBonus is the bonus if the customer stays in TT, and the FunBonus is the sum of the reservation values of all the entertainment a customer receives.

At the end of each game, the TAC scorer (on the TAC server) allocates the agent's travel goods to its individual customers optimally. The value for a particular allocation is the sum of the individual customer utilities. The agent's final score is then the value of this allocation minus the cost of procuring the goods.

For full details and examples visit `http://www.sics.se/tac/`

4.2 Functional Requirements

Design and implement an agent capable of entering the TAC-competition. The basic mechanism are provided by the DummyAgents which you can download from the TAC-website (http://www.sics.se/tac/). This DummyAgent leaves you to implement the decision-making parts of your agent, or in other words to provide your agent with bidding strategies for the various actions in your game. Designing a bidding strategy for a TAC auction context is a challenging problem. First you should consider the interdependencies. These exists between the different kinds of auctions (e.g. flights will be useless if the hotel rooms are not available); between different dates within the same kind of auction (customers have to stay in the same hotel during their entire holiday); and between the same day, same kind counterpart auctions (if the price of TT for a certain day is high, the customer can change to SS for the same day). Second, bidding involves uncertainty. For example, flight prices start randomly and change continuously in a random fashion; one randomly selected hotel auction closes anywhere between minute 4 and minute 11. Third, trade-offs exist in bidding. For example, in flight auctions, if an agent buys all the flight tickets very early, it may fail to buy the necessary hotel rooms that the flights require.

The main TAC-server on `agentcities.cs.bath.ac.uk` and a number of testing servers will always be available so that you can test your implementation. If you want to test the strength of your implementation against some of the best previous participants in the TAC-game you can connect to the SICS server. You can do this by typing the following:

```
java -jar tacagent.jar\
     -host tac.sics.se\
     -agent <agentname>\
     -password <password>
```

4.3 Non-functional Requirements

- The program must be entirely written in Java.
- The program should be written in a object-oriented fashion. Methods consisting of more 30 lines of code will be looked at with suspicion.
- The design should be as modular as possible, allowing reuse of code. Furthermore, the design should anticipate possible extensions of the software.
- The program should use the software provided by TAC[2] to connect to the server. Any other implementation will not be accepted.
- The behaviour of your agent should be different from that exhibited by the TAC DummyAgent.
- The program should work correctly on the BUCS-system.
- Only the standard Java API may be used, in addition to the API provided by the TAC-team.

4.4 Installing

- Download TAC AgentWare for Java - Beta 6
- Unzip the archive
- In a Unix/Linux environment: compile by typing “make”
- In a Windows: compile using: javac se/sics/tac/*/*.java” followed by ”jar cfm tacagent.jar AWManifest.txt se/sics/tac/*/*.class”
- In the file “agent.conf” replace “host=localhost” with “host=agentcities.cs.bath.ac.uk”

[2]`http://www.sics.se/tac/`

- Remove the ’#’ before “#agent=agentName” and “#password=password” and replace agent-Name and password to the values given to you.
- Run the DummyAgent using: java -jar tacagent.jar

4.5 The Report

The documentation[3] that accompanies your agent submission should contain:

- A project description
- Two (one each) completed personal report forms (a template provided on the CM30174 web site)
- A detailed description of the bidding strategies you implemented
- A justification for the decisions made
- A critical analysis of your implementation
- Directions for further improvements
- The source code

5 Assessment

5.1 Conditions

The coursework will be conducted in pairs. The pairs will be determined by the unit lecturer when students’ unit choices have been finalized. Attention is drawn to the University rules on plagiarism, on the guidelines on group coursework in the Departmental Handbook (section 11.3.4) and the Departmental QA-procedures for group formation and group assessments (To be distributed). The coursework can be worked on during the tutorial sessions. The coursework also involves work during one’s own study time.

5.2 Estimated Workload

The coursework will count for 25% of your final mark. The average amount of time each of you are intended to spend on this coursework is 15 hours.

5.3 Marking

Thirty percent of your coursework mark is based on your agent’s average score following a number of test runs against the other submissions. An average score between 0 and 1000 yields 12 marks, between 1000 and 1500 will give you 15 marks, between 1500 and 2000 adds 18 marks to your total, with an average score between 2000 and 3000 you obtain 21 marks. The full marks (30 marks) is given for an average score equal to or higher than 3000. The other

[3]There should only be one document for each pair.

70 marks come from the assessment of your accompanying documentation. The following is a guideline so that you are aware of what you may typically need to do for the documentation:

- 1st Class. The deliverables reach the standard for a 2:1 student. In addition, the documentation demonstrates deep understanding of the problem and pulls in ideas from background reading. The documentation demonstrates a clear appreciation of other areas of knowledge and sources of expertise that might be brought to bear on the coursework. The documentation demonstrates innovative ideas for implementing bidding strategies.
- 2:1. The coursework idea is clearly well understood. Evidence of the identification of background reading sources is shown. The description of the bidding strategies and the decisions made is precise and succinct. Alternative end-points for the coursework have been considered and identified. Evidence is supplied for trying various bidding strategies for your agent.
- 2:2. The students started to expand/modify the original coursework idea to suit their own skills and understanding. Some evidence of background reading is shown. The description of the bidding strategies and justification of decisions made demonstrates effort to make the agent as intelligent as possible in the time-frame of the coursework.
- Third. The students have hardly expanded on the supplied DummyAgent and the description of bidding strategies demonstrates a lack of understanding of the code that was given to them. Very little effort is made to come up with more advanced strategies.

The personal reports will be used to identify the effort of the individual team members. If team members fail to agree on the work they have done, a viva may be necessary to identify personal contributions. Where team effort is not equally balanced the individual marks may be adjusted.

Feedback will be given in the form of individual written comments.

5.4 Deadlines

The coursework is published on **Thursday 6th October**. Copies of the description will be made available on the website `http://www.cs.bath.ac.uk/~jap/CM30174/` and from the departmental office (after the lecture).

The deadline for this coursework is **Monday 7th November, 12pm**.

Your team should submit the documentation as described herein and a floppy, securely bound in with the documentation. The floppy should contain your source code in one directory and the jar-file obtained by compiling[4] your code in another directory. Make sure that the jar-file is named uniquely using the format `user-id1+user-id2.jar`.

The final TAC competition used for your agent's score will be held during the week of **Monday 21st of November**.

[4]Details on how to obtain the jar file can be found in the installation section above.