Mertacor: A Successful Autonomous Trading Agent

Panos Toulis*†, Dionisis Kehagias†‡, Pericles A. Mitkas†‡,
†Aristotle University of Thessaloniki, Dept. of Electrical and Computer Engineering,
54124, Thessaloniki, Greece, +30-2310996390
‡Centre for Research and Technology Hellas, Informatics and Telematics Institute,
57001, Thermi, Greece, +30-2310464160
mertacor@olympus.ee.auth.gr

ABSTRACT

In this paper we present the internal architecture and bidding mechanisms designed for Mertacor, a successful trading agent, which ended up first in the Classic Trading Agent Competition (TAC) of 2005. TAC provides a realistic benchmarking environment in which different trading agents compete with each other in order to best satisfy their clients' preferences and maximize their total profit. The "travel game" scenario of TAC involves three types of auctions; a) continuous one-sided that sell flight tickets, b) ascending multi-unit auctions for booking hotel rooms, and c) continuous double auctions for entertainment tickets. For each one of these types, we describe the techniques deployed by Mertacor. In flight auctions prices are updated according to a random walk process, thus the accurate prediction of the next update is not feasible. A key element of agent behavior in these auctions is its ability to accurately deduce the specific time in an auction, at which bidding will be profitable. In order to deal with the uncertainty due to price fluctuations in flight auctions and to provide our agent with an efficient decision mechanism, we have designed the *z-heuristic* framework. The goal of *z*-heuristic is to figure out when the price assumes its minimum value and recommend bidding at that moment. In the case of hotel auctions, Mertacor used fuzzy reasoning in conjunction with rule-based reasoning in order to predict the closing prices of hotel rooms, when historical data from past auctions are available. In order to bid efficiently in entertainment auctions, we have designed a bidding strategy whose goal is to preserve a pre-specified long-term profit. We finally present and discuss the results of agent benchmarking in the TAC Classic game.

Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Artificial Intelligence-Intelligent Agents

General Terms

Algorithms, Performance, Design, Economics, Experimentation.

Keywords

Agent-mediated E-commerce, Trading Agents, Auctions.

*This author is a student

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AAMAS'06, May 8-12, 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005...$5.00.

1. INTRODUCTION

Autonomous agents that participate in electronic marketplaces represent a new paradigm of automated transactions, in which computer programs are faced with the challenge to effectively increase their revenue without human intervention. The successful performance of agents in trading scenarios, measured by the profit or savings they materialize, may introduces a great potential for agent-mediated e-commerce in general and electronic auctions in particular.

This paper presents Mertacor, a trading agent that ended up first in the finals of international Trading Agent Competition for 2005 (TAC-05). The latter provides a competitive environment where a number of autonomous trading agents compete to each other in order to procure travel packages on behalf of a number of customers in online simultaneous and interrelated auctions of different types. In this paper we describe Mertacor's architecture and internal structure, as well as the details of the bidding algorithms designed to deal with the different types of auctions in the context of TAC.

The TAC (http://www.sics.se/tac) provides a competitive trading environment in which, each participating agent operates with the goal of assembling travel packages on behalf of eight clients. Each package refers to a 5-day period travel and consists of a round-trip flight, a hotel reservation and tickets for three different entertainment events. The clients have separate preferences over the arrival and departure dates, the type of hotel and entertainment events they wish to visit, which are randomly assigned to each client at the beginning of the game. The objective of each agent is to maximize the total satisfaction of its clients. In the TAC simulation environment, all three kinds of commodities (flights, hotels and entertainment tickets) are sold in simultaneous online interrelated auctions of three different types, running over a game, which lasts for 9 minutes. TAC has been designed so that no optimal strategy that always wins may exist. On the contrary, the agents that participate in the competition need to deploy tradeoffs between the offered goods in order to best satisfy their clients' preferences. The TAC supported auctions are described in the following.

There is only one Airline Company that sells tickets in single seller continuous one-sided auctions, which close at the end of the game. Each auction sells tickets for a particular day and direction, whereas an unlimited number of seats are available. The first agent that submits a bid in an auction buys the auctioned ticket for its current price and the auction clears. Once a bid is submitted it cannot be withdrawn. Prices in flight auctions are updated according to a random walk process. The initial price is a uniformly distributed variable in the range [250,400]. Each price update is then perturbed every 10 seconds by a value uniformly drawn from the following intervals:

- [-10, x(t)] if x(t) > 0
- [x(t), 10] if x(t) < 0
- [-10,10] if x(t) = 0.

The bound $x(t)$ is calculated by the following equation:

$$x(t)=10+(t/9{:}00)(x-10)\ , \quad (1)$$

where x is a hidden random variable drawn uniformly on [-10, 30] and 9:00 denotes the auction duration (9 minutes).

There are two hotels in which clients can stay between the arrival and departure dates: “Tampa Towers” (TT) and “Shoreline Shanties” (SS). TT is more comfortable, clean and preferable than SS, thus it is expected to be more expensive. Hotel rooms are traded in standard ascending multi-unit 16^{th} price English auctions, which close at randomly determined times in the last 8 minutes of the game. In each auction 16 rooms are offered for each combination of hotel and night. A hotel auction clears and matches bids only once, when it closes. On clearing, hotel rooms are allocated to the 16 highest offers with all competitors paying the price of the lowest winning bid. Price quotes are only generated once per minute, on the minute.

Entertainment tickets are traded in continuous double auctions (CDA), which are held between the participants during the game. Each agent holds a randomly chosen number of tickets from the beginning of the game and may operate as either a seller or a buyer. Trade may occur at any time during the game and participants are allowed to continuously update their prices, i.e. buyers may update their offered prices and sellers their ask prices. Entertainment ticket auctions clear continuously. On clearing, bids match immediately. A bid that does not completely match remains standing in the auction.

The score that the agent receives at the end of each game is calculated as the utility minus the expenditure costs. The utility function in its general form is:

Utility = 1000 – *travelPenalty* + *hotelBonus* + *FunBonus*,

Apart from tackling with the utility optimization problem, TAC participants need to also deal with many uncertainty factors introduced by the different nature of each auction types and the interrelations that hold between them. For example, the agents need to acquire flight tickets, hotel rooms and entertainment tickets so that are all consistent with the preferred arrival and departure dates on behalf of their clients. Agents should also take compound decisions, including how much and when to bid. In order for an agent to operate efficiently it has to monitor its environment and collect sufficient information about the market conditions and the behavior of its opponents.

Since the beginning of the competition in 2000, the TAC problem attracted many participants from different countries and organizations. ATTac-2000 agent [7], [8] made the first key contribution to this challenging area. The intricacies of the game were clarified and attacked in a systematic way. ATTac-2000 was the first agent who won the TAC. The notion of *marginal utility* was then recognized to play an important role in the TAC game framework. In fact it has been proven that bidding marginal values in sequential auctions with deterministic prices is an optimal strategy and a fairly satisfactory one in the TAC environment [2]..

Since the first TAC, teams concentrated their efforts mainly on developing with effective ways of price prediction. A survey of agent prediction methodologies deployed in TAC-02 may be found in [10]. Due to space limitations we only refer here to those approaches that most influenced our work. A novel price prediction method was designed for agent SouthamptonTAC [3] by the application of fuzzy techniques. Walverine [1] on the other hand provided an analytical approach relied on the principles of a competitive economy. Another approach was proposed by whitebear [9], which simply but also interestingly enough, used average prices for prediction. LearnAgents [6] introduced an alternative multiagent architecture as opposed to most competitors' single agent approaches.

Although all of the above efforts generated good results, they had not traced thoroughly the details of the flight ticket price perturbation process. Having realized that the hidden variable x is the most significant parameter in flight auctions, Mertacor's flight bidding component provides a formula that calculates the probability of the fact that x lies on a particular continuous interval. This approach is innovative in the context of TAC flight auctions.

In this paper we describe the implementation details of Mertacor and illustrate the key contributions of our work. In particular, we first present the internal architecture of our agent, in Section 2, which proved to be very stable since the beginning of the competition. An additional feature of the described architecture is that it easily allows rapid changes in agent's internal bidding mechanisms due to its high degree of modularity.

The next key contribution of this paper is the description of a novel decision framework, called *z-heuristic*, designed in order to deal with the problem of optimal bidding in flight auctions.

Section 4 describes the mechanism we have deployed in order to deal with price prediction in hotel auctions. In particular, we adopted a previously known fuzzy approach, originally developed for agent SouthamptonTAC [3], which was a competent in TAC-01 and TAC-02. Further to this approach, we have also slightly modified its functionality, by adding a new set of rules that improve the results of the fuzzy reasoning module.

Another contribution of this paper, described in Section 5, is the introduction of a new bidding strategy for continuous double auctions, which are realized in the form of entertainment auctions in the context of TAC.

Section 6 provides an outline of the benchmarking results taken from TAC-05, as well as independent experiments, while Section 7 concludes the paper and outlines future work.

2. MERTACOR ARCHITECTURE

Mertacor's architecture, which is shown in Figure 1, adopts a typical modular design that supports synchronous intercommunication. Three main units (Flight, Hotel and Entertainment Auction Unit) undertake the responsibility to concurrently handle information related to the three different types of auctions running on the TAC server. Typically, this information includes the prices of the goods that are auctioned off. This configuration adheres to the typical abstract agent architecture, where the three units play the role of Mertacor's sensor components. An additional sensor unit, which also receives game information pertaining user preferences and running auction id*s* is the *Game Info* component.

All input information perceived by the sensor components of Mertacor goes to the *Bidder* module, which deploys the three bidding mechanisms developed for each of the aforementioned auction types into three separate modules, one for each type of auction. Once *Bidder* calculates the new bids it sends them back to the TAC server. Thus, *Bidder* implements the main "effector" components of Mertacor. Since there is one direct communication channel to the TAC server, which is served through a TCP connection, it is critical to allow all three parts of Bidder to synchronously send information to the TAC server. For this reason, the three components of *Bidder* are implemented as threads. For instance, the *Hotel Bidder* component instantiates a thread process that places bids in hotel auctions every minute. *Bidder* also provides central control services that are activated when special conditions of emergency arise (e.g. a bid is rejected by the server).

Mertacor relies on the execution of a Linear Programming (LP) model. This is implemented as a separate module used not only to provide the optimal goods allocation but also the marginal value of the traded goods and the shadow prices of the model's constraints [5]. In what follows we provide thorough descriptions of the internal bidding mechanisms of Mertacor with respect to the three separate modules.

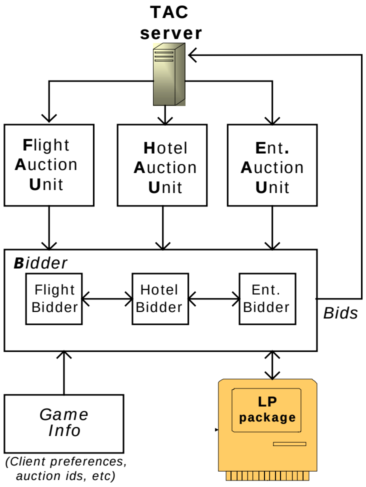

Figure 1. Overview of Mertacor's architecture

3. BIDDING IN FLIGHT AUCTIONS

According to TAC flight auction rules, price perturbations at time t are related to the value of the random variable *x(t),* which is given by Equation (1). The ranges, from which perturbations are drawn, are determined by the sign of the bound $x(t)$. The main objective of our analysis is to figure out the time at which *x(t)* changes sign. This time guarantees a profitable buying. This holds because *x(t)* is a linear function depending on time and on the hidden variable *x*. Excluding extreme cases, this means that *x(t)* either decreases or increases, according to *x*. Moreover, prices are updated either from D_1 =[-10, $x(t)$] or D_2=[$x(t)$, 10]. Since prices are initially updated from D_1, if *x(t)* increases in a timely fashion, it is easy to show that prices will most likely increase. On the other hand, if *x(t)* decreases, it is likely for it to change sign. In this case, the interval D_1 decreases, thus prices decrease accordingly. When *x(t)* changes sign, prices are updated from D_2. Since *x(t)* changes linearly, at this point it is negative and close to zero. Hence, prices will start rising again. Thus, bidding at this point will potentially result in a profitable transaction.

The goal of the decision framework we describe in the remaining Section, is to make an estimation about the real value of the *x* parameter and therefore about the change of the variable *x(t)*. For this purpose, we exploit the price updates that are revealed to agents every 10 seconds. These updates provide us with *low bounds,* i.e. with real numbers denoted as B_i that satisfy the inequality $x \geq B_i$.

Knowing the point where *x(t)* changes sign is very useful. By clustering these bounds into a vector and calculating the probability of appearance of this specific vector we can construct an interval where the hidden parameter lies, with any pre-specified probability

3.1 The *z-heuristic* Decision Framework

In this section we present the z-heuristic, a framework used for estimating the time at which the prices of flight tickets reach their minimum value. For simplicity we introduce the *y* variable as follows:

$$y = x - 10 \, ,$$

where *x* is the hidden variable appeared in Equation (1). Since *x* is a random variable uniformly drawn on [-10, 30], it is clear that *y* is also a random variable uniformly distributed in [-20, 20] Each flight auction lasts for 9 minutes and posts a price update every 10 seconds. Therefore, the total number of price updates during a flight auction is N=54 . Thus, Equation (1) can be rewritten as:

$$x_i = 10 + i \cdot \frac{y}{N}, i \in [1, N] \qquad (2)$$

Firstly we will define the point at which $x(t)$ changes sign, as a *corner*. For this purpose we introduce the following definitions.

Definition 1. Consider two updates i_1 and i_2. We define that update i_1 is *earlier* than i_2 (or equivalently, i_2 is *later* than i_1), iff: $i_1 < i_2$.

Definition 2. We define as *corner* i_c of a flight auction the latest update for which we know that price perturbations are drawn from [-10, x_i].

Definition 3. We define as *low bound B* the maximum real number for which we know that $y \geq B$.

Our goal is to calculate the corner. Lemma 1 provides a way to find a candidate low bound that may lead to a corner.

LEMMA 1. *If* δp_i *is the price variation at the* i^{th} *update for which is* $i \leq i_c$, $B_i = (\delta p_i - 10) \cdot \frac{N}{i}$ it holds that

$$y \geq B_i \qquad (3)$$

Due to space limitations we skip the proofs of this and the following Lemmas. If $B_i > B$ then we have found a new low bound of the flight auction. This also gives us useful information about the corner of the flight auction, according to the following Lemma.

LEMMA 2. *If a flight auction has low bound B, then the corner* i_c *is given by* $i_c = \left\lfloor 10 \frac{N}{|B|} \right\rfloor$, if $B \leq -10$, or $i_c = N$, otherwise.

It is important to clarify the distinction between the corner that is derived from auction data and the *real corner* of the flight auction, i.e. the update at which the bound x_i really changes sign. Our goal is to calculate the real corner. Since this variable changes linearly, the values of perturbations are drawn from [x_i, 10], after x_i changes from zero to a negative value. In this case, prices will obviously rise because x_i is much less than 10. Thus, the price changes are positive and generally high. This explains why there is a steeply decreasing stage for flight auctions whose y parameter is below –10. After that stage an increasing one follows. The prices are reduced until the real corner of the flight auction is "crossed" and then they increase. Therefore, it is a good strategy to buy just when an update is estimated to be close to the real corner.

It is likely that the corner calculated by Lemma 2 is different than the real one. For example, let us consider a flight auction that has a real corner i_{rc} = 36. Setting $x_{i_{rc}}=0$ in (2), we get y=-15. In order for a new low bound to occur, price changes should be noticeably big, according to inequality (3). If this is not the case, the only information we are able to know about y is that $y \geq -20$. Therefore, B = -20 and the flight auction's corner is 27, according to Lemma 2. Buying at the 27th update implies that we will miss 9 updates at which price is very likely to decrease. This is of course undesirable. Thus, instead of buying at the auction corner, we define a new buying criterion based on the following Lemma:

LEMMA 3. *If* $B_i=(\delta p_i-10)\cdot\frac{N}{i}$ *, B is auction's low bound and* i_c *is the flight auction corner, the following statements cannot be true at the same time:*

- $B_i < B$
- $i > i_c$
- *the real corner has been crossed*

*for the i*th *price update.*

Therefore, if a flight auction has bound B, corner i_c and for the i^{th} update the following two statements are true:

- $i > i_c$
- $B_i \geq B$,

our agent assumes that the real corner has been crossed, and decides to buy the tickets needed.

Normally, in a flight auction many successively increasing bounds B_i may come up at the i^{th} update. We define as *evidence couples* any couple of the form (B_i, i), and use them to estimate the value of variable y. By recording all evidence couples found during an auction we can provide the following definition.

Definition 4. Bound evidence (*BE*) is a vector of the form $BE=\left[(b_1,k_1) \quad (b_2,k_2) \quad \cdots \quad (b_n,k_n) \quad T\right]$, where $b_i=\frac{B_i}{20}$ and $i \in$ 1,2,…n.

Let us also define variable $s = y / 20$. The variable T is an integer equal to the last update of the flight auction.

It is very useful to calculate the probability $p(BE|s)$ of the fact that a specific bound evidence occurs in an auction, when the value of the variable s is known. This is provided by the following Lemma.

LEMMA 4. *If s=y/20, the probability of occurrence of the evidence couples* (b_1, k_1) *and* (b_2, k_2) *with* $b_2 > b_1$ *and* $k_2 > k_1$ *is*

$$\Upsilon(k1,\ k2,\ b1,\ s) = \left\{\prod_{i=k_1+1}^{k_2-1} \frac{(N+i\cdot b_1)}{(N+i\cdot s)}\right\} \cdot \frac{k_2\cdot(s-b_1)}{(N+k_2\cdot s)} \tag{4}$$

By generalizing Lemma 4, the probability $p(BE|s)$ can be calculated, using (4), by the following equation:

$$p(BE \mid s) = \prod_{i=1}^{n-1} \Upsilon(k_i,k_{i+1},b_{i-1},s) \cdot \prod_{j=k_n+1}^{T} \frac{(N+j\cdot b_n)}{(N+j\cdot s)} \tag{5}$$

Equation (5) describes $p(BE|s)$ as a function *g(BE, s)* of the hidden variable s.

Since $y \in [-20,20]$, it is $s \in [-1,1]$. From probability $p(BE \mid s)$, we can calculate p(s| BE) which will give us an estimation of s and thus y. Using Bayesian probabilities we can write

$$p(s \mid BE) = \frac{p(BE|s)\cdot p(s)}{p(BE)} \tag{6}$$

Setting s = s_1 and s_2 successively in Equation (6) we create two new equations from which we derive that $p(s_1| BE)p(BE| s_2) = p(s_2| BE)p(BE| s_1)$, or:

$$p(s_1|BE)g(BE,s_2)=p(s_2|BE)g(BE,s_1) \tag{7}$$

Now let us consider a continuous subset $S_1 \subseteq [-1, 1]$. Keeping s_2 fixed and having $s_1 \in S_1$ by integrating equation (7) we have:

$$p(s \in S_1|BE)g(BE,s_2)=p(s_2|BE)\int_{S_1} g(BE,s) \tag{8}$$

Assume that $S_2 = [b_n, 1]$. Since B_n is actually the low bound of the flight auction and b_n is the normalized low bound, it is:

$$p(s \in S_2 \mid BE) = 1 \tag{9}$$

From Equations (8) and (9) for $s_2 \in S_2$ we have:

$$p(s \in S_1|BE)=\frac{\int_{S_1} g(BE,s)}{\int_{S_2} g(BE,s)}$$

If we consider $S_1 = [b_n, z]$ we can gradually increment z so that $p(s \in S_1 \mid BE)$ becomes equal to a fixed predefined value p. We chose this value to be 0.8. With this algorithm we can find a range [b_n, z] in which the parameter s lies, with probability 0.8. This value of the z is called *z-point***.** The reason for applying such a methodology was motivated by an experimental analysis whose results are graphically described in Figure 2.

In Figure 2 we can see the relationship between the bound of the y parameter found until the 20th update for 50,000 auctions and the real value of y. Obviously, real y values tend to gather around and below the line y=B. This is the reason we use the range [b_n, z] in our z-heuristic framework**.**

Having constructed a range in which we assume with sufficient evidence that the hidden parameter s lies, we can make an estimation of its real value. In order to accomplish this, we defined the *sign evidence (SE) vector* as a vector of the signs of all auction price updates. Following a procedure similar to the one used for the calculation of probabilities of occurrence of the *BE* vector we can calculate $p(SE)$ and $p(SE|s)$.

The estimation of s is enabled through the following equation**:**

$$\hat{s}=\frac{1}{z-b_n}\cdot\int_{S_1} s\cdot\frac{p(SE|s)}{p(SE)}ds \tag{10}$$

Thus, the estimation of y is simply derived by the equation:

$$\hat{y}=20\cdot\hat{s} \tag{11}$$

Equations (10) and (11) provide an approximation of y that can be used in conjunction with statistical methods, in order to enable an estimation of the price change from the arbitrary update i_a to the

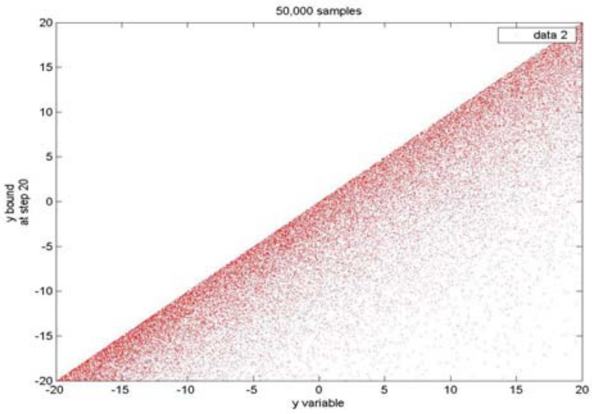

Figure 2. Low bounds found until the 20th update for 50,000 games. Samples are gathered around line *y=B* which provides a good estimation of the real *y* value.

later update i_b, where $i_b < i_a$. By considering the mean value in the range [-10, x_i], we receive:

$$\widehat{\Delta p}(i_a, i_b) = \hat{y} \cdot \frac{(i_a+i_b)\cdot(i_b-i_a+1)}{4\cdot N}$$

Finally, we determine which values of the above estimations play a significant role in the bidding process. For this reason, we define the following decision criteria, considering a flight auction with current price *CP*, initial price *IP*, current update *i* and low bound *B*:

1. Is T>52 ?
2. Is CP<160$?
3. Is *Corner* crossed?
4. Is $\hat{y}>4$ AND $B>-3$ AND $CP>IP$?
5. Is B>7 ?

3.2 Flight Auctions Analysis

Each of the previously defined criteria is potentially met according to the current status of the auction parameters. A *decision variable*, which is an integer $d = 0,\ldots,5$ represents one of the previously defined decision criteria. A positive value of d triggers the bidding mechanism, 0 means that no criterion is satisfied. For example, if $d = 5$, criterion 5 is met, hence prices will most probably skyrocket. By using these criteria, our agent may opt to postpone a bid submission or fragment the total quantity for which it intends to bid.

The results of auction analysis are reported in Table 1. This table shows the evolution of a real flight auction during the first 9 updates.

The flight auction starts off with an initial value of $327. At update no.3 an increment of $9 gives a new low bound B = -18 and the corner becomes 30. At the 5th update a price increment by 10 gives a new bound B = 0 and now the corner becomes 54, which implies that until the end of the auction, price fluctuations will be drawn on [-10,x_i]. Since bound is zero, it will be $y \geq 0$ and y is estimated to the value 8.99. At this time criterion 3 is fulfilled, therefore the decision variable changes to value 3. This indicates that current time is a good time to buy the auctioned ticket. The final price was ended up to $388, thus the recommended decision on bidding results to a $35 profit for our agent.

Table 1. Flight Auctions Analysis table (9 updates)

i	i_c	Initial price	Current price	Last bound	Z-point	Decision	Y est.	Buying Price
1	27	327	327	-20.0	11.88	0	-4	327
2	27	327	332	-20.0	11.60	0	-4.28	332
3	30	327	341	-18.0	15.49	0	-0.73	341
4	30	327	343	-18.0	15.29	0	-0.49	343
5	54	327	353	0.0	17.34	4	8.99	353
6	54	327	352	0.0	17.34	4	8.86	352
7	54	327	345	0.0	17.19	4	8.63	345
8	54	327	350	0.0	17.19	4	8.78	350
9	54	327	361	6.0	18.25	4	12.29	361

4. HOTEL BIDDING STRATEGY

At the beginning of the game, there is little or no information about hotel prices. For this reason we use the following procedure in order to calculate the initial bids to submit. Denoting (*arrival day*, *departure day*) a user preference over the arrival and departure dates, there are 10 possible such travel preferences: (1,2), (1,3), (1,4), (1,5), (2,3), (2, 4), (2, 5), (3, 4), (3, 5) and (4, 5). Four out of ten preferences include an in-flight at day 1, three at day 2, two at day 3 and one at day 4. Thus, we can define a parameter that indicates how likely for an in-flight day is to occur among client preferences. Let us call this parameter *statistical importance factor* of in-flights for the day *i*:

$$s_{in,i} = 0.5 - 0.1i, i \in [1,4],$$

Similarly, the statistical importance factor of out-flights for the day *j* is:

$$s_{out,i} = 0.1j - 0.1, \ j \in [2,5]$$

Our goal is to estimate hotel closing prices. For this purpose, we use the only information, which is available in the beginning of each game, i.e. the initial flight prices. Our methodology uses these prices to estimate the average hotel room demand, which indicates how far from the statistical balance the game will evolve. The probability that a client will be staying for *i* days is equal to: $p(i) = 0.5 - 0.1 \cdot i$. Thus, the *mean trip duration* equals to $\sum_{i=1}^{4} p(i)\cdot i = \sum_{i=1}^{4} (0.5-0.1\cdot i)\cdot i = 2$ days and hence the average demand for hotel rooms per game is:

$$2\frac{rooms}{client} * 8\frac{clients}{agent} * 8\frac{agents}{game} = 128\frac{rooms}{game}$$

Since 16 rooms are sold in each of the 8 hotel auctions, the *hotel room supply* is equal to the average demand. It is reasonable to assume that in games with demand close to the average demand, a balanced bidding activity will be noticed. Otherwise, the game may become unstable and the hotel prices skyrocket.

Let us assume that the eight competitors need I_i in-flight tickets at day *i* and O_j out-flight tickets at day *j*. Total in-flight and out-flight demand will be $I = \sum Ii = 64$ and $O = \sum Oi = 64$, respectively, since we have 64 clients and one in-flight and out-flight preference for each client. Let us also define the *relative* in-flight and out-flight ticket demand as $\varepsilon_i = I_i / I$ and $\mu_i = O_j / O$, respectively, with $i \in [1,4]$, $j \in [2,5]$. If the absolute hotel room demand for day i is H_i, then it is easy to prove that the following equations hold: $H_1 = I_1$, $H_2 = I_1 + I_2 - O_2$, $H_3 = -I_4 + O_4 + O_5$ and $H_4 = O_5$, or equivalently:

$$\mathbf{H} = \mathbf{R} \cdot \mathbf{F}, \tag{12}$$

where $\mathbf{H} = (H_1 \quad H_2 \quad H_3 \quad H_4)^{\mathrm{T}}$,

$$\mathbf{R} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

and $\mathbf{F} = \begin{pmatrix} I_1 & I_2 & I_3 & I_4 & O_2 & O_3 & O_4 & O_5 \end{pmatrix}^{\mathrm{T}}$. Equation (12) maps absolute flight demand to absolute hotel demand.

The estimation of the hotel demand for each day is calculated by the following equation:

$$\hat{\mathbf{H}} = \mathbf{R} \cdot \hat{\mathbf{F}}, \tag{13}$$

where:

$$\hat{\mathbf{F}} = 64 \cdot \begin{pmatrix} \hat{\varepsilon}_1 & \hat{\varepsilon}_2 & \hat{\varepsilon}_3 & \hat{\varepsilon}_4 & \hat{\mu}_2 & \hat{\mu}_3 & \hat{\mu}_4 & \hat{\mu}_5 \end{pmatrix}^{\mathrm{T}},$$

and $\hat{\varepsilon}_i$ $\hat{\mu}_j$ are the relative demand estimates for in-flight and out-flight tickets respectively. In order to calculate $\hat{\varepsilon}_i$ we use the following equations:

$$\hat{\varepsilon}_i = f_{in,i} / \sum_{k=1}^{4} f_{in,k} \text{ and } f_{in,i} = s_{in,i} \cdot (400 - IP_i) / 150,$$

where IP_i is the initial in-flight ticket prices for day i. The $\hat{\mu}_j$ coefficients are calculated in a similar manner.

After estimating hotel room demand we create initial bids by using look-up tables with historical data from previous games. These tables provide mappings between closing prices and hotel room demand estimates. As real information about market demand is being revealed in the game, our agent exploits the fuzzy reasoning method [4] developed for SouthamptonTAC [3] for price prediction. In our implementation, we used the prediction p combined with the mean value m of the closing prices denoted at the previous 14 games. We use a Linear Programming module in order to calculate the optimal allocation of the hotel rooms, when the closing price is estimated to be a linear conjunction of the previous prediction p and the mean value m. In particular, the closing price estimation, denoted I_{LP}, is calculated in a heuristic manner by the following rules:

IF $p \geq 2m$ **THEN** $I_{LP} = p$;
ELSE IF $p \leq m/2$ **OR** $p \geq (3/2)m$ **THEN** $I_{LP} = (2/3)p + (1/3)m$;
ELSE $I_{LP} = (1/2)(p+m)$;
END IF

During the last two minutes of the game, when uncertainty about the market is reduced, using the above estimation of hotel closing prices increases bidding efficiency

5. ENTERTAINMENT BIDDING STRATEGY

Entertainment auctions are continuous double sided auctions, where agents receive new price quotes every 30 seconds and bids are processed continuously. Although the entertainment auctions adhere to the continuous double auction protocol applied in the stock market, they formulate a more simplified auction environment for two reasons. First, the number of eight participants is significantly lower than the ones met in a typical stock market. Second, the agents remain adherent to their initial plans about acquiring the desired tickets. This is not always the case in a real stock market, where traders may deploy totally unpredicted bidding behaviors.

Mertacor's bidding mechanism in entertainment auctions is based on a simple and consistent algorithm that aims to achieve a long-term profit at the end of the auction.

Our selling strategy in the entertainment auctions is implemented by the procedure *MertacorSellStrategy* illustrated in Figure 3. We describe its functionality in what follows.

```
PROCEDURE MertacorSellStrategy
 1:  FOR each entertainment ticket in possession
 2:      Assign a pre-specified value to target;
 3:      mean ← getMeanValue();
 4:      M ← A*target + B*mean;
 5:      IF M ≤ (1/2)*target OR M ≥ (3/2)*target THEN
 6:         M ← relocateM();
 7:      END IF
 8:      V ← calcVal();
 9:      IF V < Vo THEN V ← Vo; END IF
10       profit ← ask– 2*V;
11       Mt ← w(t)*M;
12       Ra ← M;
13       Rb ← 3*M/2;
14       Rc ← rand(M/2, M);
15       IF profit ≥ Mt-Ra THEN sellTicket();
16       ELSE IF profit ≥ Mt-Rb THEN ask (Mt-Ra+2*V);
17       ELSE ask (Mt - Rc +2*V);
18       END IF
19   END FOR
```

Figure 3. The selling strategy deployed by Metacor in the entertainment auctions.

The procedure iterates over all tickets that the agents possesses. The *target* variable represents the long-term average profit that the agent aims at achieving. This is set to a pre-specified value for each of the entertainment tickets. Typical values for *target* lie on [5, 12]. The variable *mean* is set equal to the current mean value of profit gained over the previously completed transactions. This is calculated by the function *getMeanValue()* in row 3. Variable M, which is given by the following equation:

$$M = A \cdot target + B \cdot mean \tag{14}$$

is used to determine the range of the profit sought. After experiments, the weights A and B were chosen to be A=0.7 and B=0.3. Thus, our agent seeks for a profit highly influenced by the value of *target*. In order to keep our agent adherent to this goal, we impose (row 5) the variable M to only take values in the range [(1/2)·*target*, (3/2)·*target*]. For this reason, we use function *relocateM*() (row 6), which returns values in the range [min {*target*, *mean*}, max {*target*, *mean*}]. In order to calculate current ticket's valuation V, our agent calls function *calcVal()*, in row 8, which makes use of the LP model. Since we are interested in always selling higher than a determined reserve price V_o, if $V<V_o$ we set $V=V_o$ in row 9. If we denote *ask* as the ask price, then Mertacor calculates the profit it would make in a potential transaction by using the equation

$$profit = ask - 2V \tag{15}$$

This preserves our agent's goal, which is to keep $E\{profit\}=target$, where the $E\{.\}$ operator denotes the mean value. In row 11 we define a new variable $M_t = w(t)M$ that introduces a time-depended low bound for the desired profit. The $w(t)$ function is graphically represented in Figure 4. The peak that appears in the middle of the game (4.5 minutes) represents the maximum profit sought at that time. The demand for profit decreases and then remains constant at a relatively low value as the game reaches its end. Next, we define three variables, namely R_a, R_b and R_c, in rows 12, 13 and 14 respectively. Function *rand()* in row 14 returns a number uniformly distributed in the range [M/2, M]. These three variables determine, along with the variable M_t the ranges of an acceptable profit. In order to decide when to bid, our agent uses the three decision rules shown in rows 15-17. According to these rules, Mertacor sells the ticket it currently possesses if the profit to be achieved is M_t-R_a or above. This is done by function *sellTicket()*, in row 15. Otherwise, it asks for those prices that will result to a profit equal to or bigger than M_t-R_a or M_t-R_c. Our agent submits its ask price (derived from (15)) for both cases, by invoking the *ask()* function in rows 16 and 17, respectively. Since $R_c \le R_a$ the expected profit will be at least equal to $(w(t)\text{-}1)M$.

From Figure 4 we can see that $w(t) \geq 1$ for most of the time. During the last 30 seconds of the game $w(t)$ is fixed to value 0.8 and the profit becomes -0.2M, which is a negative value. Thus, if M has a big value, a transaction occurred during the last 30 seconds of the game, will lead to a big loss. In addition to this, transactions close to the middle of the game are not so likely to happen, since the quantity $(w(t)\text{-}1)M$ is relatively big. This will lead to a decrease of the overall average profit, and thus M as it is derived from Equation (14). On the other hand, if M is small, transactions that lead to a positive profit are more likely to happen, while negative profit transactions will result to a small loss. The aforementioned mechanism is highly adaptive to changes in the market environment, since when a big profit is assumed the number of transactions is reduced and vice versa. The buying bidding strategy of Mertacor is completely symmetrical to the selling strategy.

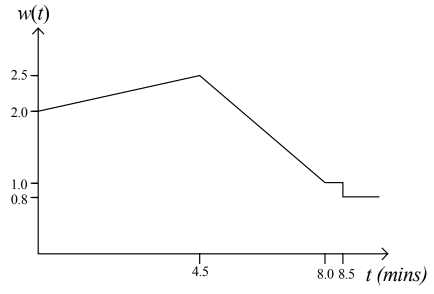

Figure 4. The *w*(*t*) function plotted over time (in minutes). It indicates a time-dependant bidding attitude

6. RESULTS

In the 6th TAC, eleven agents participated from the beginning of the competitions. Among these only eight managed to end up to the finals. The agents were evaluated according to the average score they gained at each round. The score of each agent is calculated by the server when a game finishes and it is equal to the utility minus expenditure costs. For each round the average score determines the performance metric for all agents.

TAC-05 consisted of 5 rounds. In the first (qualifying) round of TAC all eleven agents participated in 600 games, running for almost two weeks in a constant 24-hour basis. Our agent ranked fourth, gaining a score of 3918.45, which was 240.33 below the top score achieved by agent whitebear05. In this round Mertacor employed a completed version of the Flight Auctions strategy, while it used a naïve bidding algorithm for hotel auctions and a greedy strategy for entertainment auctions. Next, in the seeding round 688 games were played using the same server configuration. At the beginning of this round we changed our hotel bidding mechanism to the one presented in Section 4 and preserve it until the finals. We also used for the fist time the CDA strategy described in Section 5. We set the parameter *target* in Eq. (14) to value *target* = 12, i.e. our agent was seeking for a big profit. In this round Mertacor improved its overall performance. It finished third with score 4033.32, managing to reduce its distance from the top score agent whitebear05 to 135.29. In the semi-final round ten participants were able to compete in 56 games. Mertacor improved many aspects of all bidding algorithms and mostly fixed many bugs. Thus, it retained its performance by gaining a score of 4023.88.

Eight out of the ten competitors who participated in the semi-finals were invited to the finals held on August 3rd in Edinburgh, Scotland. Table 2 shows the scores of the eight finalists. In the final round all agents played in 80 games, concurrently run in two servers (40 games in each server). Compared to the semi-final round, Mertacor improved its entertainment bidding strategy, by fine-tuning the parameter in the entertainment stategy. In particular, we lowered the value of the *target* parameter to 5. This resulted in both lower profits and losses. This intervention proved to significantly improve our agent's performance who eventually ranked first.

Apart from the overall benchmarking results provided by TAC, we have also conducted an additional analysis of the agents' performance regarding only the entertainment auctions, because fine tuning the *target* parameter in the entertainment bidding algorithm proved to be the key element that increased our agent's performance. In the context of our experiments, we measured the profits that the agents made in their transactions, in all the games of the final round. The results of this analysis are shown in Fig. 5. For each agent we measured the average profit collected in buying, selling and both types of transactions. Fig. 5 shows that the proposed algorithm is particularly efficient in buying transactions. Mertacor performed almost as well as its competitors in selling transactions. In the latter case the best performing agent is Dolphin. Noticeably, Mertacor and whitebear05 are the only agents that managed to receive a positive profit when participating in buying transactions, with Mertacor outperforming all agents. This resulted in an overall better performance of Mertacor in the entertainment auctions.

Table 2. Score table for TAC-05 final round

Agent	Average Score	Zero Games
Mertacor	4126.49	0
Whitebear05	4105.68	0
Walverine	4058.90	2
Dolphin	4022.79	0
SICS02	3972.29	0
LearnAgents	3899.24	0
e-Agent	3451.25	0
RoxyBot	3167.64	10

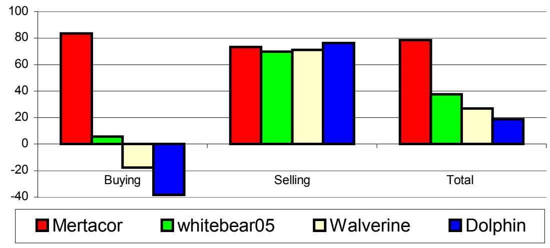

Figure 5. Average Profit collected by the four top-scoring agents in the final round when they participated in buying, selling and in both types of transactions

7. CONCLUSIONS

This paper presented the details of Mertacor, a trading agent which took part in TAC-05 competing against eleven agents in a simultaneous interrelated auctions environment. We analyzed our agent's architecture and provided the details of the internal bidding mechanisms. Specifically, we presented the bidding strategies deployed by our agent per each of the three different types of auctions involved in the TAC game. In flight auctions we tackled with the decision problem of choosing the best time to bid in the auction, i.e. the time that leads to a maximum profit. In order to calculate the time at which bidding is more profitable, we have developed a methodological framework called z-heuristic based on probabilistic reasoning. The outcome of z-heuristic is then used by our agent in conjunction with a set of simple bidding rules in order to decide when to bid. Next, we explained the price prediction method adopted for efficient bidding in the hotel auctions. This method comes from previous related work, and it was slightly enhanced in order to meet our agent's design requirements. Although we developed a new technique for price prediction in hotel auctions, we applied it on Mertacor only at the qualifying round, while we then switched to the adopted methodology, which we kept until in the finals rounds for the sake of performance stability. In order to deal with bidding in double-sided entertainment auctions we developed a strategy with the goal to achieve a long- term profit specified by a target value. This strategy proved to be robust and easily adaptable to market fluctuations, especially when Mertacor participated in buying transactions. Our agent, which exploited a combination of strategies, exhibited outperformed performance in the context of the TAC.

Future work involves further improvement of the presented techniques and continuation of our experiments.

8. ACKNOWLEDGEMENTS

We would like to thank the TAC organizing teams, University of Michigan and SICS for their technical support before and during the competition.

9. REFERENCES

[1] Cheng, S.-F., Leung, E., Lochner, K. M., O'Malley, K., Reeves D. M., Schvartzman L. J., and Wellman, M. P. Walverine: A walrasian trading agent. In *Proceedings of the Second international Joint Conference on Autonomous Agents and Multiagent Systems* (Melbourne, Australia, July 14 - 18, 2003). AAMAS '03. ACM Press, New York, NY, 465-472.

[2] Greenwald, A, and Boyan, Bidding Under Uncertainty: Theory and Experiments. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*. (Jul. 2004), 209-216.

[3] He, M., and Jennings, N. R. Designing a Successful Trading Agent: A Fuzzy Set Approach. *IEEE Transactions on Fuzzy Systems, 12,* 3 (Jun. 2004), 389-410.

[4] Kandel, A., and Langholz G. *Fuzzy Control Systems*, CRC Press, 1993.

[5] Kolman, B., and Beck, R. *Elementary Linear Programmig with Applications*. Academic Press Inc. (London), 1995.

[6] Sardinha J. A. R. P., Milidiú R. L., Paranhos P. M., Cunha P. M., Lucena C. J. P. An Agent Based Architecture for Highly Competitive Electronic Markets. In *Proceedings of the 18th International FLAIRS Conference,* (Clearwater Beach, Florida, USA, May 16-18, 2005).

[7] Stone, P., Littman, M. L., Singh, S., and Kearns, M. ATTac-2000: An Adaptive Autonomous Bidding Agent. *Journal of Artificial Intelligence Research 15* (2001)*,* 189-206.

[8] Stone, P., Schapire, R., Littman, M., Csirik, J., and McAllester, D. Decision-theoretic bidding based on learned density models in simultaneous, interacting auctions, *Journal of Artificial Intelligence Research, 19*, (2003), 209-242.

[9] Vetsikas, I. A. and Selman, B. 2003. A principled study of the design tradeoffs for autonomous trading agents. In *Proceedings of the Second international Joint Conference on Autonomous Agents and Multiagent Systems* (Melbourne, Australia, July 14 - 18, 2003). AAMAS '03. ACM Press, New York, NY, 473-480.

[10] Wellman, M. P., Reeves D. M., Lochner, K. M., and Vorobeychik, Y. Price prediction in a trading agent competition. *Journal Of Artificial Intelligence Research*, *20* (2004) 1-17.

[11] Wellman, M. P. Wurman, P. R. Oapos Malley, K. Bangera, R. Lin, S.-d. Reeves, D. Walsh, W. E. Designing the market game for a trading agent competition. *IEEE Internet Computing, 5*, 2 (2001), 43-51.

[12] Zimmermann, H. J. *Fuzzy set theory and its applications*, Kluwer, Boston, 3rd edition, 1999.