WhiteDolphin: A TAC Travel Agent

P. Vytelingum, R. K. Dash, I. A. Vetsikas and N. R. Jennings
School of Electronics and Computer Science,
University of Southampton,
Southampton, SO17 1BJ, UK
{pv,rkd,iv,nrj}@ecs.soton.ac.uk

Abstract

In this paper, we detail our WhiteDolphin agent that was designed for the Trading Agent Competition (TAC) Travel game. Specifically, we employed the multi-layered IKB framework to design our strategy, and describe the intricate cogs involved at the different layers in this complex decision-making process. We focus, in particular, on WhiteDolphin's strategic behaviour when bidding in the different types of auctions involved in the game, and how the information and knowledge required to support the complex decisions made is gathered and inferred respectively. Finally, we empirically analyse our agent by considering its performance in the 2006 competition where it ranked third.

Introduction

The last decade has seen an upheaval in the nature of electronic commerce as software agents with principally economic motivations have emerged as rational players that are capable of autonomous and flexible actions to achieve their objectives. Such agents are needed because today's electronic trading markets[1] allow access to an abundance of information that enables trading agents to be more informed and respond more efficiently than humans could ever hope to. To be effective, however, these agents often need to be endowed with intelligent and complex profit-maximising strategies. The design of such strategies is still a significant research challenge and one that we deal with in this paper.

In more detail, the Trading Agent Competition (TAC) provides an excellent benchmark platform to develop software agents that are capable of complex and autonomous decision making on behalf of human owners. TAC Travel, one of the two different strands of the Trading Agent Competition, looks at how software agents can be designed to be automated travel agents building holiday packages for a number of clients. The game attempts to address real-life issues such as limited supply of hotel rooms, changing flight prices or the changing demand of hotel rooms as the deadline (beginning date of the holiday) approach. Thus, we are required to develop autonomous software agents that are capable of profit-maximising behaviours through effective trade-offs (Vetsikas & Selman 2003) (since we have conflicting goals of utility maximisation to satisfy clients' demand and cost minimisation) and risk management (as the agent has to factor in the uncertainties about the current and future state of the TAC marketplace into its decision making).

Against this background, in this work, we develop a strategy for one such software agent, which we name *WhiteDolphin* (to signify its origins from the Whitebear and Dolphin agents). In particular, we adopt the multi-layered IKB framework[2] (Vytelingum *et al.* 2005) to assist the design of our strategy. Thus, we will describe in detail the components of the strategy within the different layers (the information, the knowledge and the behavioural layer) of the IKB. Specifically, we will first look at the high-level behaviour of our agent in the behavioural layer, covering what to buy, when and at what price. Next, we will consider the knowledge required for such behaviour and how such knowledge is inferred using techniques such as machine learning and price forecasting. Finally, at the information layer, we consider the low-level information that can be observed in the TAC marketplace and which is required in the knowledge layer. Our strategy differentiates itself from others in a number of ways as we shall see in this paper. Now, the main novel aspects of this strategy lie first in its flight prediction, and how it uses this prediction effectively, and, second, in its use of a state of the art trading strategy for Continuous Double Auction to trade in the entertainment auctions.

In this paper, we begin with a general description of the TAC Travel game. We then describe our WhiteDolphin strategy and how it was engineered using the IKB framework and then empirically evaluate it. Finally, we conclude and highlight future work towards improving our strategy.

The TAC Travel Game

This game involves a number of software agents competing against each other in a number of interdependent auctions (based on different protocols) to purchase travel packages

[1] An electronic trading market is defined as an online institution with an exchange of resources or services using a currency as the trading means. Such markets range from auctions, to supply chains, to barter systems.

[2] The IKB framework assists the strategy designer at the conceptual level, by designing the strategy at different complexity levels. Such a framework is particularly useful when designing a strategy for a complex game as the TAC Travel.

over a period of 5 days (for the TACtown destination) for different clients. In more detail, in a TAC Travel game (that lasts 9 minutes), 8 agents are required to purchase packages for up to 8 clients (given their preferences). To do so, they compete in 3 types of auctions:

1. *Flight auctions*. There is a single supplier for in-flight and out-flight tickets over different days, with unlimited supply, and ticket prices updating every 10 seconds. Transactions occur whenever the bid is equal to or greater than the current asking price of the flight supplier.
2. *Hotel auctions*. There are two hotels at TACtown, namely Shoreline Shanties (SS) and Tampa Towers (TT), with TT being the nicer hotel, and each hotel having 16 rooms available over 4 different days. Thus, there are 8 different hotel auctions (given the 2 hotels and rooms being available for 4 different days). Hotel rooms are traded in 16th-price multi-unit English auctions, whereby the sixteen highest bidders are allocated a room for a particular day in a particular hotel, and at the end of every minute except the last, a hotel auction randomly closes, and the 16th and 17th highest price of each hotel auction that is still open is published.
3. *Entertainment auctions*. There are three types of entertainment in TACtown, namely a museum, an amusement park and a crocodile park, and 12 different entertainment auctions (for the three type of entertainment tickets for each of the four days). At the beginning of the game, each agent is randomly allocated 12 entertainment tickets tradeable in the different multi-unit continuous double auctions (CDAs) which clear continuously and close at the end of the game.

The WhiteDolphin Strategy

Given this background on TAC Travel, our objective is to design a trading strategy for an autonomous software agent participating in such a game. We develop the strategy by using the IKB framework, adopting the multi-layered approach (see Figure 1) as discussed in the introduction. We now describe the strategy within the different layers prescribed by IKB; namely the *behavioural layer* (BL), which describes the higher-level decision-making process, the *knowledge layer* (KL), which describes the knowledge required for the decisions in the BL and the *Information Layer* (IL), which describes the low-level information the agent has about itself and what it has observed in the market.

The Behavioural Layer

In the BL, we design the strategy at a high-level and, specifically, we look at the broad issues associated with the bidding behaviour. These can be summarised as follows:

1. What item to bid for?
2. How much to bid for?
3. When to bid?

We address the first issue by considering the *optimal plan* (see Definition 1). Thus, the agent always bids for the exact set of items (flight tickets, hotel rooms and entertainment tickets) required for the optimal plan, querying the optimal plan from the KL every 10 seconds. As a hotel auction closes every 60 seconds, the set of items available to the agent is further constrained and the optimal plan has to be recalculated. We address the other issues by considering the different auction formats.

Definition 1 ***Optimal Plan.*** *The optimal plan is the set of travel packages, for 8 different clients, that yield the maximum profit, given the clients' preferences and the cost of the packages.*

Definition 2 ***Marginal Profit*[3].** *The marginal profit of a hotel room (in a particular hotel on a particular day) is the decrease in the agent's total profit if it fails to acquire that room. Thus, the marginal profit of a hotel room that is not required in the optimal plan is 0.*

First, we consider the eight flight auctions. Given the manner in which the flight prices update, it is possible to predict the trend of the price update. Such a trend is queried from the KL. If the trend suggests a decrease in price, the BL then queries the predicted lowest ask price of the flight auction, and a bid is placed in that auction when that minimum is reached, if such flight tickets are required in the optimal plan. Conversely, if an increasing trend is predicted in a flight auction, we face a trade-off between acquiring all the tickets in such an auction immediately at the current lowest price, and waiting in case the agent does not manage to acquire the *scarce* hotel rooms required in the optimal plan, which could make the flight tickets redundant (since they are no longer required in the optimal plan and represent a loss). We implement the trade-off by spreading our bids in the flight auctions over the remaining length of the TAC game. For example, if 4 tickets are required from a particular flight auction with an increasing trend, we would buy a single ticket every minute over the next 4 minutes, rather than buying them all immediately.

Next, we have the eight hotel auctions, with a random one clearing (and closing) every minute. Thus, every minute, as the optimal plan changes, we update our bid in those auctions that are yet to close. Now, there is uncertainty associated with being able to acquire all the items required in the optimal plan, particularly at the beginning of the game. Because a bid in a hotel auction can only be replaced by a higher bid, and because the optimal plan typically changes during the game, it does not pay to bid too high for an item at the beginning of the game. This is because that item might no longer be required in the optimal plan as the game progresses, thus might be acquired and not used. Given this, our agent does not bid for a hotel room at its marginal profit (see Definition 2), but rather bids low at the beginning of the game (at a fraction, inferred from previous simulation, of its marginal profit) and gradually increases its bid for a room towards its marginal profit as the game progresses, bidding

[3]The marginal profit described here is broadly similar to the marginal value used in (Cheng *et al.* 2003).

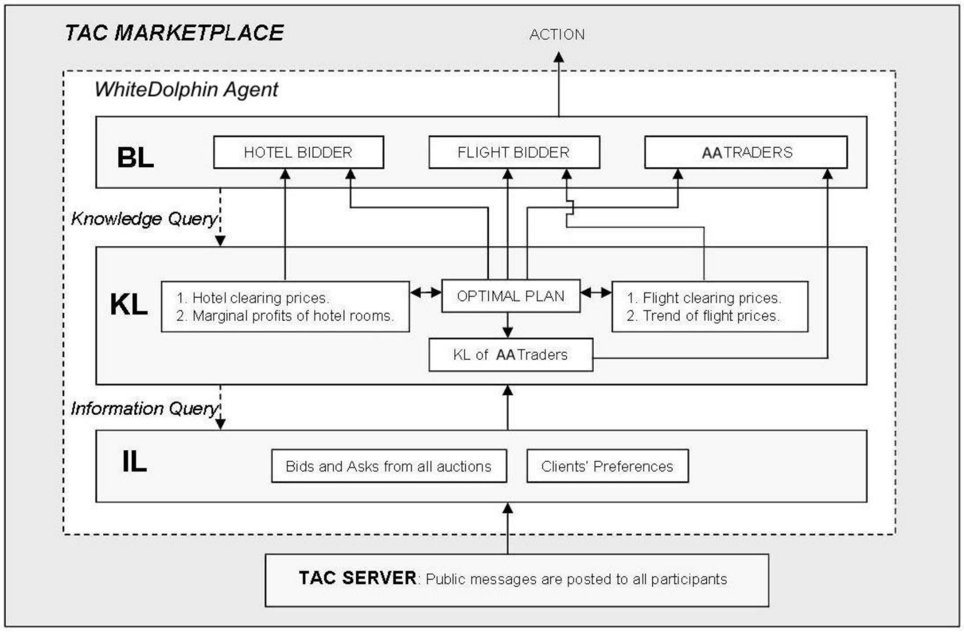

Figure 1: Structure of the WhiteDolphin Strategy for the TAC Travel game.

its marginal cost after the 7th minute before the last hotel auction closes.

Finally, we have the twelve entertainment auctions. Here, we use a modified version of the AA strategy (Vytelingum 2006) (designed for the standard CDA model). In particular, we have 12 AA traders that each bid in one of the entertainment auctions for the tickets required in the optimal plan. The agent further instructs the AA trader to buy *cheap* in auctions that do not influence the optimal plan, and sell as *high* as possible all tickets that could potentially be sold to other agents for a higher profit than what the agent would get by giving them to its clients. Because the AA agents are also designed using IKB, with the multi-layered approach (see Figure 2), this makes it easier to port the AA strategy to WhiteDolphin as a multi-layered black box with the inputs at the information layer and the outputs at the behavioural layer. The WhiteDolphin agent thus delegates the entertainment bidding to the AA agent, instructing it on the number of entertainment tickets it needs to buy or sell in its auction. In this paper, we do not describe the AA strategy in detail, but rather simply use its multi-layered design to simplify the description of our strategy in a similar manner as it simplifies its design.

Against this background, we now consider the knowledge required for the general bidding behaviour of our agent.

The Knowledge Layer

This layer accummulates the knowledge required for the behaviour adopted by the WhiteDolphin agent. As we can see in the middle box of Figure 1, the behavioural layer requires different knowledge about the game:

1. The optimal plan
2. The flight prices prediction
3. The hotel prices prediction
4. The marginal profit (see Definition 2) of each hotel room
5. The entertainment ticket price prediction

In this Subsection, we look at each type of knowledge in detail, and how it can be inferred.

The Optimal Plan: Here, we principally require the optimal plan (see Definition 1) which is given as the solution to an optimisation problem[4]. Specifically, the agent searches for the plan that maximises its profit, which is the total utility of the packages less their estimated cost. The utility of a package is determined by the clients' preferences, which is queried from the IL. Furthermore, the optimisation problem is constrained by different requirements of a feasible package, for example a client needs to stay in the same hotel for the duration of his/her stay or the client is required to stay in a hotel during the length of his/her stay (Wellman *et al.* 2002), with additional constraints imposed as hotel auctions close.

We modelled our optimisation problem on (He & Jennings 2004)[5]. Assuming that we wish to find the optimal plan when considering only flights and hotels, we have a

[4]We use ILOG CPLEX 9.0 to solve the optimisation problem, with a solution typically found within a few milliseconds.

[5]We improved upon their approach by considering that entertainment tickets can be auctioned off when it is beneficial to do so. This allows the agent to determine which entertainment tickets to buy or sell.

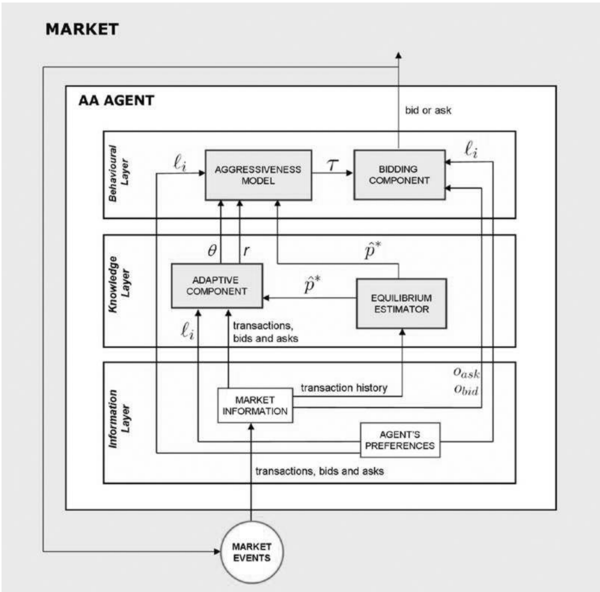

Figure 2: Structure of the AA Strategy for Continuous Double Auctions.

solution set, $\mathcal{S}$, $f_{i,j} \in \{0,1\}$, $\forall j = \{0,..,19\}$, of 20 packages for each client $i \in \{0,..,7\}$. The principal motivation for such a simplification of the problem is that the problem is computationally more tractable (given that the optimiser is run several times per second). Furthermore, the price volatility of the entertainment packages causes the optimal plan to change too frequently (possibly every ten seconds) which make the bidding process less reliable (as the agent would be changing its bids as frequently to reflect its knowledge of the optimal plan). The problem is then to find the number of goods to buy (and sell in the entertainment auctions) for each of the 28 different auctions. We denote these by $BUY[j]\ where\ j \in \{0,..,27\}$ and $SELL[j]\ where\ j \in \{0,..,27\}$, subject to the following constraints:

$$
\begin{aligned}
BUY[j] &\in \{0,..,8\}\ \forall j \in \{0,..,7\}\ (flight)\\
BUY[j] &\in \{0,..,4\}\ \forall j \in \{8,..,15\}\ (hotel)\\
BUY[j] &\in \{0,1\}\ \forall j \in \{16,..27\}\ (entertainment)\\
SELL[j] &= 0\ \forall j \in \{0,..,15\}\\
SELL[j] &\in \{0,1\}\ \forall j \in \{16,..27\}
\end{aligned}
$$

Because the demand at the current bid and the supply at the current ask price in the entertainment auctions are unknown (with only the quotes made public), we limit the number of goods to buy or sell in this particular type of auction to 1. Furthermore, we avoid a plan where the agent is required to hold an excessive number of the same hotel room (and adopt a less risky approach by spreading more of its bids across the different hotel auctions) by constraining the number of the same hotel rooms in our solution. Against this background and denoting the entertainment bonuses[6] by $E_{i,j} \in (0,200)\ \forall j = \{0,..,11\}$, owning an entertainment ticket as $e_{i,j} \in \{0,1\}$, $\forall j = \{0,..,11\}$ for each client $i \in \{0,..,7\}$ and the utility of each package $f_{i,j}$ as $u_{i,j}$, we maximise the following objective function (which is the profit):

$$
\max \sum_{i=0}^{7}\sum_{j=0}^{19}(f_{i,j} \times u_{i,j}) + \sum_{i=0}^{7}\sum_{j=0}^{19}(e_{i,j} \times E_{i,j})
- \sum_{j=0}^{27}(BUY[j] \times price[j]) + \sum_{j=16}^{27}(SELL[j] \times price[j]) \quad (1)
$$

To meet the requirements of a feasible package for the TAC game, we impose a set of constraints (similar to those in the optimisation problem modelled by (He & Jennings 2004)) on our optimisation problem. While the exact formulation is given by He & Jennings, we summarise the constraints as follows:

(i) The number of hotel rooms required must be less or equal to the number of room the agent owns or intends to acquire.

(ii) The entertainment tickets must be used within the time of the client's stay.

(iii) Each client, i, has only one valid package, $\sum_{j=0}^{19} f_{i,j} \leq 1$.

(iv) The number of entertainment tickets used must be less or equal to the number of tickets the agent owns and intends to acquire less the number it intends to sell, $\sum_{i=0}^{7} e_{i,j} \leq OWN[j+16] + BUY[j+16] - SELL[j+16]$ for entertainment auction $j \in \{0,..,11\}$ where $OWN[j]$ is the number of goods the agent owns. This differs from He & Jennings's model as we consider buying and selling entertainment tickets in our model.

(v) The number of flight tickets required must be less or equal to the tickets the agent owns or intends to acquire.

(vi) For each client, each type of entertainment ticket can be used only once.

The solution to our optimisation problem is then the optimal plan defining the different optimal packages, $f_{i,j}$ and the different number of goods to buy ($BUY[j]$) or sell ($SELL[j]$) in each auction. Given how we model our optimisation problem, the next issue is the pricing of the goods. A myopic approach would be to consider the current prices, but this would ignore the fact that prices changes, and an agent can gain by knowing whether prices will rise or fall, whether to buy immediately or wait for better opportunities. Indeed, our strategy considers predicted prices at when it intends to buy the goods given its behaviour (see Subsection above on the behavioural layer). We next describe the approach we take to predict these prices.

[6]The hotel bonuses are already factored in the utility for packages within set $\mathcal{S}$ where we consider the two types of hotels.

The Flight Price Prediction: As shown in Figure 1, among other tasks, the KL needs to predict the flight prices. This knowledge is required to calculate when it is best to buy the flights (given the number of flights required in the optimal plan) given the strategy of WhiteDolphin (prescribed in the BL). Furthermore, these predicted prices are fed into the optimiser in the KL to calculate the optimal plan.

Now, within TAC, the flight prices are set by TACAIR, which has an unlimited supply of seats. The price of a flight at time t, $p(t)$ is governed by a stochastic process in which an *unknown* variable x sets the overall trend in the flight price (e.g. whether it is an increasing or decreasing function). The variable x is drawn uniformly from the interval $[-10, 30]$. Furthermore, there is a second variable $x(t)$, which is dependent on x as given below:

$$x(t) = 10 + \frac{t}{54} * (x - 10) \tag{2}$$

where t is equal to the time elapsed since the game began in s. It is this variable which then determines the perturbation $\delta(t)$ in the flight price by setting the distribution from which $\delta(t)$ is drawn. More specifically,

$$\delta(t) \sim \begin{cases} U[-10, x(t)] & x(t) > 0 \\ U[x(t), 10] & x(t) < 0 \\ U[-10, 10] & x(t) = 0 \end{cases} \tag{3}$$

The perturbation is then added to the previous flight price, thereby giving the current flight price $p(t) = p(t-1) + \delta(t)$.

In more detail, Figure 3 demonstrates the behaviour of the above equations, on expectation, for different values of x . As can be observed there are three main types of behaviour which the flight price can exhibit (apart from the trivial one of staying constant at $x = 10$), dependent on the interval in which x falls. Specifically, the following behaviours are observed:

1. $10 < x \leq 30$: The expected flight price **constantly increases** from its initial value. The rate of increase, $\frac{dp(t)}{dt}$ is dependent on x, with a higher value of x leading to a greater rate.
2. $0 < x < 10$: The expected flight price **constantly decreases** from its initial value with $-\frac{dp(t)}{dt}$ being higher when x has a lower value.
3. $-10 \leq x \leq 0$: The expected flight price initially decreases till it reaches a certain **minimum** before $t = 540s$, after which it rises. The point at which it reaches its minimum is after $t = 270s$ and gets closer to $t = 540s$ as x increases. Furthermore, the extent to which the expected flight price decreases is higher when x is higher.

Now, these three behaviours will elicit different responses from the flight strategy. However, before discussing the flight strategy in detail, we will first explain how we estimate the parameter x.

Let $m(t)$ denote the difference between the current flight price (at time t) and the initial flight price. Then $m(t) = \sum_{i=0}^{t} \delta(t)$ where $\delta(t)$ is dependent on x (as per equations 3 and 2). Then, given a current perturbation $m(t)$, we want to estimate x so as to predict future perturbations. Using Bayes' rule, this can be expressed as:

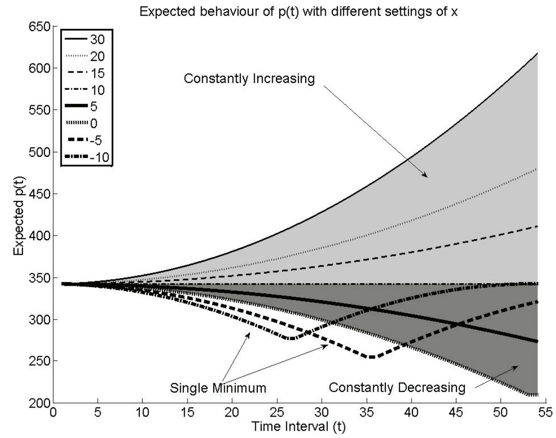

Figure 3: Figure showing different behaviours of the price $p(t)$ with different x at time intervals given by $ceil$(time elapsed in s/10). The behaviour is shown on expectation and thus does not have any noise element within it.

$$P(x|m) \propto P(m|x)P(x)$$

In order to calculate the probabilities above, we first discretise x into 40 unit intervals since this number of intervals is high enough to enable us to make accurate predictions but low enough for the rapid computation required within TAC. Also, it can be deduced that $P(x) = 1/40$ for all the discretized values of x. Thus, we only need to estimate $P(m|x)$ which is equivalent to the probability of the sum of t variables, $(\delta(1), \delta(2) \ldots, \delta(t))$, drawn independently from different uniform distributions, $f_1(\delta(1)|x), f_2(\delta(2)|x), \ldots, f_n(\delta(t)|x)$. The resultant pdf of $m|x$ is a convolution of the pdfs of $\delta(t)|m$:

$$f(m|x) = \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \Big[f_1(\delta(1)|x), \ldots \\ f_n((m - \delta(1) \ldots - \delta(t-1))|x) \Big] d\delta(1) \ldots d\delta(t)$$

We then use the result from (Bradley & Gupta 2002) in order to derive the following expression for the pdf $f(m|x)$:

$$f(m|x) = \frac{\sum_{\vec{\eta} \in \{-1,1\}^t} \left(x + \sum_{j=1}^{t} (\eta_j a_j - c_j) \right)_+^{t-1}}{(n-1)! 2^t \prod_{j=1}^{t} a_j} \tag{4}$$

where $\vec{\eta} = (\eta_1, \eta_2, \ldots, \eta_t) \in \{-1, 1\}$ (i.e. $\eta_j = \pm 1$ and $y_+^n = (max(y, 0))^n$). The value of a_j and c_j are then calculated as follows:

$$\begin{array}{lll} c_j = 0.5(x(j) + 10), & a_j = 0.5(10 - x(j)) & \text{if } x(j) < 0 \\ c_j = 0, & a_j = 10 & \text{if } x(j) = 0 \\ c_j = 0.5(x(j) - 10), & a_j = 0.5(x(j) - 10) & \text{if } x(j) > 0 \end{array}$$

Hence, upon obtaining a certain $m(t)$, we can now calculate $P(m|x)$ and therefore $P(x|m)$ for each of the 40 discretized regions. The final variable required in order to predict the expected clearing price p_c, which is the price at which WhiteDolphin would buy the flight ticket, is the time at which the ticket will be bought. This time is however dependent upon the strategy employed in the behavioural layer. In fact, as discussed above, there are three patterns that $p(t)$ can follow dependent upon the interval that x falls into. These in turn elicitate different strategies as explained below:

1. $10 < x \leq 30$: There are in fact two strategies which we employ corresponding to the following two cases:
   (a) If $10 < x \leq 30$, we then immediately buy all the required tickets.
   (b) On the other hand, if $10 \leq x < 20$, we stagger the times (over a maximum interval of $80s$) at which the tickets required in the optimal package are bought. The difference between the times at which the ticket is bought is smaller as x is closer to 20. This is in order to allow some flexibility in the optimal package bought since flight tickets tend to constitute a fairly large part of the cost incurred when calculating a travel package.
2. $0 < x < 10$: Since the price of the ticket constantly decreases, we buy the ticket in the last few seconds on the round as the price will be close to its lowest then.
3. $-10 \leq x \leq 0$: In this case, a minimum occurs in the price of the ticket. We therefore need to predict the time at which this minimum occurs and buy around this minimum price.

Thus, in order to find which of the above strategy to employ, we first group the probabilities $P(x|m)$ for each of the following intervals , $[-10,0], (0,10], (10,20], (20,30]$ as follows:

$$P(i \leq x \leq j|m) = \sum_{k=i}^{j} P(x_k|m)$$

Furthermore, we can estimate the most likely value of x, $\overline{x}$, as:

$$\overline{x} = E(x|m) = \sum_{x=1}^{40} xP(x|m)$$

If the group with the highest probability contains $\overline{x}$, then this sets the strategy which we employ when buying the ticket. In more detail, if $\overline{x} > 0$, then we can expect $x(t) > 0 \quad \forall t$. Thus, $E(\delta(t))$ is given by:

$$E(\delta(t)) = t(\overline{x} - 10)/108$$

Now, note that each of the three strategies covered by $\overline{x} > 0$, provide a time at which a ticket (demanded in the optimal package) needs to be bought. Thus, if the expected clearing time is t_c, then the expected clearing price as calculated at time t, is given by:

$$\begin{aligned} p_c &= p(0) + m(t) - \sum_{i=1}^{t} \frac{i(\overline{x}-10)}{108} + \sum_{i=1}^{t_c} \frac{i(\overline{x}-10)}{108} \\ &= p(0) + m(t) - \sum_{i=t}^{t_c} \frac{i(\overline{x}-10)}{108} \\ &= p(0) + m(t) - \frac{(t_c - t)((t_c + t)\overline{x} - 20)}{2} \end{aligned} \tag{5}$$

In strategy $1(a)$, $t_c = t$ since the ticket is bought immediately. In strategy $1(b)$, different tickets are bought at different times, which we limit to $80s$ from the time t_{iden} at which $\overline{x}$ is classified as being in the interval $[10, 20)$. Thus, $t_{iden} < t_c < t_{iden} + 8$ and the cost for each ticket can be calculated accordingly. For strategy $1(c)$, $t_c = 52$ since the ticket is bought within the last $20s$.

Now, in the case that $\overline{x} < 0$, it can be observed that $x(t) > 0$ till the time, t_{min} at which the minimum point occurs. Now, from Equation 2, $t_{min} = \frac{540}{10-\overline{x}}$ which can be substituted into Equation 5 to give the expected clearing price.

Having thus explained how WhiteDolphin performs the flight price prediction, we now go on explain the method it employs to predict hotel prices.

The Hotel Price Prediction: Based on our analysis of previous TAC games, we observed that a good prediction of hotel prices was possible based on mainly the current ask price. However, because hotel prices are updated every minute, at the beginning of the game, a prediction based on past history was desirable. Thus, we considered a simple prediction method by considering the average[7] clearing price for each of the 8 hotel auctions.

The Marginal Profit of the Hotel Rooms: Given our optimal plan that uses our predicted prices and determines which hotel rooms to buy, the next problem is to profitably bid for these rooms. Specifically, if buying a particular room does not make the agent more profitable, it must not make a purchase. Thus, we calculate the marginal profit $MP_{i,j} \forall hotel\ auction\ i \in \{8,..,15\} \forall j \in \{0, N_i\}$ (where N_i is the number of hotel rooms required for a particular auction i as $(MP_{i,N}, MP_{i,N-1}, ..., MP_{i,0})$ (with $MP_{i,N} = 0$ since we are considering the optimal plan from which it is not profitable to deviate). Here, we are considering the next best package if a particular hotel room in the optimal plan cannot be acquired. The drop in profit then represents the marginal profit of that hotel room.

The Entertainment Ticket Price Prediction: WhiteDolphin delegates the task of bidding in entertainment auctions to AA agents by specifying what ticket to buy or sell as given

[7] Note that we do not consider the minute at which the hotel auction clears. We could improve our prediction technique by predicting the clearing price at each minute for each hotel auction. However, this means considerably more training data over 64 different cases (the clearing price for each hotel at each minute), and because the long duration of the TAC game limits the training set, a good prediction might not be possible given such an approach.

by the optimal plan. Now, in the optimal plan, we require the predicted clearing price, $price_i$, in each entertainment auction i (see Equation 1). This clearing price will lie between the bid and the ask price in the auction. Now, when the bid-ask spread Δ_i (the difference between the bid b_i and the ask a_i) in auction i is small, the clearing can be fairly accurately predicted, even by considering a mean. However, when the bid-ask spread is large (for example after the market clears or at the beginning on the game), a mean is typically not very accurate (for example if we have a bid of 10 and an ask of 90, it is not realistic to predict a clearing price of 60 given that we know that the market tends to clear around 80). Thus, we adopt a non-optimistic approach and assume overly profitable transactions are not possible. Specifically, we predict a higher clearing price (closer to the ask) for the buyer and a lower clearing price (closer to the bid) for the seller:

for buyer,

$$price_i = \begin{cases} 200 & \text{if } a_i = 0 \\ a_i & \text{if } \Delta_i > max \\ b_i + \left(0.5 + 0.5(\Delta_i/max)^\beta\right) a_i & \text{otherwise} \end{cases}$$

for seller,

$$price_i = \begin{cases} 0 & \text{if } b_i = 0 \\ b_i & \text{if } \Delta_i > max \\ b_i + \left(0.5 - 0.5(\Delta_i/max)^\beta\right) a_i & \text{otherwise} \end{cases}$$

where β determines how optimistic our prediction is, and is set to 0.5 in WhiteDolphin. max is some maximum bid-ask spread beyond which we should not expect any transaction (i.e. if the bid-ask spread is too high, for example at 110, with a bid at 20 and an ask at 130, we assume that the buyers or the sellers are currently not willing to transact and hence submitting very low bids or asks). We set max to 100 in our strategy. Furthermore, in the KL of the AA trader (Vytelingum 2006), because we do not have an equilibrium price[8] for these CDAs (due to changing demand and supply), we set the equilibrium price to the expected transaction price (calculated as the mean of the history of transaction prices). The KL of the AA traders in the TAC environment is otherwise similar to the traditional CDA mechanism.

Having considered the knowledge layer, we next consider the information layer.

The Information Layer

In this layer, the agent extracts all the information needed for the knowledge it requires. Indeed, it tracks information relevant to the TAC Travel game, such as the running time of the game and which auctions have closed, as well as the clients' preferences that do not change during the game. When it considers the individual auctions, the agent has to record the history of published information (bids and asks where available). In the flight auctions, the history of flight prices is required to estimate the trend, which represents vital knowledge. In the hotel auctions, the history of the publicly announced 16th highest price can be recorded up to when the auction closes. Such information can be used to estimate the clearing price of the hotel auctions in future TAC games. Finally, for the entertainment auctions, the agent has the same IL as the AA traders.

[8]The equilibrium price is that towards which transaction prices converge and which is the expected transaction price in the market.

Agent	Roxybot	Walverine	WhiteDolphin
Total Score	4179	4154	4129
Total Utility	9787	9847	9597
Total Cost	5608	5693	5467
F Cost	4615	4654	4591
H Cost	1102	1065	881
E Cost	-109	-26	-6
E Tickets#	11	12	12
H Bonus	613	598	517
E Bonus	1470	1530	1528
Penalty	-296	-281	-449
E Score	1580	1557	1534
Average Hotel Bid	170	115	50

Table 1: The TAC Travel competition results from the final day of the competition over 80 games for the top 3 agents bidding in flight (F), hotel (H) and entertainment (E) auctions.

Empirical Evaluation

In this section, we empirically evaluate our strategy by considering its performance in the 2006 competition. Specifically, we consider the final day of the competition[9] (over 80 games) as does (Lee, Greenwald, & Naroditskiy 2007) in their analysis of Roxybot, the competition winner. The result is given in Table 1. It is broken down to analyse the performance of our agent in the different types of auctions, and, in so doing, we attempt to identify any shortcoming and possible improvements.

Our strategy ranked third in the competition. Over the last 80 games, we averaged a total of 4129, 50 (1.2%) and 25 (0.6%) behind Roxybot (which ranked 1^{st}) and Walverine (which ranked 2^{nd}) respectively. When considering the top 3 strategies, we generally observe that WhiteDolphin has the lowest cost, but also the lowest utility. Now, a TAC strategy is typically an interplay of trade-offs between maximising utility and minimising the costs across the different types of auctions. Our strategy specifically favours minimising costs, going for the cheapest flights and hotel rooms. We now consider its performance in more detail by considering the metrics in Table 1.

First, we consider the flight auctions. Our total cost is lower than Roxybot and Walverine and, in this case, WhiteDolphin outperforms these strategies by 0.6% and 1.5% respectively (see the "F Cost" row on Table 1). Given our optimal plan, we observed that our strategy was particularly effective in predicting future clearing prices, and at bidding at the minima. This suggests that flight bidding is a particularly effective aspect of our strategy.

[9]Note that we do not consider the previous days because, due to technical difficulties, we could not run the latest version of WhiteDolphin on the first day.

In the hotel bidding, WhiteDolphin spends 20% and 17% less than Roxybot and Walverine respectively (see the "H Cost" row on Table 1). However, WhiteDolphin is more inclined to purchase rooms in the cheap hotel, which leads to a lower hotel bonus; as can be seen in the "H Bonus" row of Table 1, it gains an overall hotel bonus which is 15% less than Roxybot and 14% less than Walverine. WhiteDolphin typically avoids expensive hotel rooms, and, indeed, it bids considerably less (71% and 56% less than Roxybot and Walverine respectively) at an average hotel bid of only 50. To avoid the more expensive hotel rooms, WhiteDolphin is compelled to incur a higher penalty than the competition (52% and 60% more than Roxybot and Walverine respectively) for deviating from the clients' demand. On the other hand, Roxybot and Walverine spend more on hotels, going for higher bonuses and lower penalties. By aggregating the hotel costs, the penalties (assuming that penalties are necessarily incurred because of excessive hotel room prices rather than flight prices) and the hotel bonuses, we observe that Roxybot and Walverine outperform WhiteDolphin by 0.7% and 1.6% respectively.

In the entertainment auctions, Roxybot and Walverine do better than WhiteDolphin by 3.0% and 1.5% respectively[10] (see the "E score" row on Table 1). The primary reason for this is because WhiteDolphin is not as effective when choosing which entertainment tickets to buy and sell. Furthermore, though we use an efficient trading strategy, we do so in a market with sniping strategies and agents that bid their limit prices, avoiding bid shading through a bargaining process. In such cases, the choice of which tickets to auction is more fundamental than the actual trading strategy. Another factor that contributes to this, is that, overall, WhiteDolphin uses less hotel rooms in the packages that it gives to its customers, compared to RoxyBot and Walverine, which also affects the entertainment bonus, albeit to a lesser degree.

When we combine the different performances of the these agents across the three types of auctions, we see that Roxybot and Walverine, overall, outperform WhiteDolphin by 50 ($-0.6\% + 0.7\% + 1.1\% = 1.2\%$) and by 25 ($-1.5\% + 1.6\% + 0.5\% = 0.6\%$) respectively. While WhiteDolphin is more efficient in the flight auctions, Walverine is more efficient in the hotel auctions and Roxybot in the entertainment auctions.

Discussion and Conclusions

Having presented the WhiteDolphin strategy, it is now interesting to compare the strategies presented in the behavioural layer with the approaches and solutions employed by other successful TAC agents.

The flight auctions are a relatively novel piece of the TAC trading game in that the equation setting the pattern followed by the flight auctions was changed in 2005. We shall therefore compare our flight prediction strategy with those which appeared in the 2005 and 2006 competitions. Our flight strategy resembles those of Walverine (Wellman *et al.*) and Mertacor (Toulis, Kehagias, & Mitkas 2006), two highly successful strategies, in that they both had a prediction of the hidden variable x based on Bayesian updates. However, their strategies did not capitalise on the fact that the perturbations were drawn from a uniform distribution, as ours do.

In the hotel auctions, our agent bids low at the first minute and then gradually increases each bid towards the corresponding room's marginal profit, as time passes; this general idea is in fact used in some variation or another by most successful TAC agents. In fact, in (Vetsikas, Jennings, & Selman 2007), it is shown that, under certain conditions, this general behaviour constitutes a Bayes-Nash equilibrium for an individual TAC hotel room auction. To be more specific, the authors show that, for the distribution of valuations that rooms have at the various rounds of the TAC game, it is a Bayes-Nash equilibrium to bid some relatively small amount above the current price of the auction at the first round, and at the last round to bid almost equal to the marginal profit; in the intermediate rounds, the bids are increased gradually between these two extreme strategies. When it comes to the entertainment auctions, WhiteDolphin adopts a novel approach in that it uses an existing state of the art CDA strategy (see Subsection) given its limit price based on the supply and demand of tickets in each auction and the bonuses associated with each ticket. Roxybot on the other hand submits bids and asks based on historical data (of transaction prices) for the past 40 games (Lee, Greenwald, & Naroditskiy 2007) while Walverine uses the same limit price as WhiteDolphin, but simply submits this price rather than engage in a bargaining process as does WhiteDolphin.

While we can analyze the strategies for each set of auctions separately, to evaluate the performance of the whole agent it is advisable to experiment with different combinations of strategies. To this end, in (Reeves *et al.* 2005) an experimental methodology is used by the Walverine team to reduce the strategy space that needs to be explored in order to select the best combination of strategies. Another approach to reducing this space is presented in (Vetsikas & Selman 2003) and (Vetsikas & Selman 2005). At each step, the strategy space is explored across one dimension, by varying the strategy for one auction only, while keeping all others fixed among all the agents, and then selecting the observed best strategy for the next step. This process is repeated until the best combination of strategies is found.

To summarise, in this paper, we presented the WhiteDolphin strategy for the TAC Travel competition. We described how we strategise through different trade-offs across the different auctions (given the conflicting objectives of maximising utility, while minimising costs) and different techniques of predicting future trading prices. Our empirical analysis shows that our strategy favours minimising cost to maximising utility compared to the other successful strategies. WhiteDolphin represents one extreme of the TAC Travel strategy space, while the other extreme would be a strategy which maximises its utility by minimising its penalty, going for the hotel bonuses and maximising entertainment bonuses, but incurring a high cost.

From our analysis of how WhiteDolphin performs in the different auctions with respect to Roxybot and Walverine,

[10] This is 1.1% and 0.5% respectively, if the differences are denoted as a percentage of the total scores.

we observed that it performs better in the flight auctions and poorer in the hotel and entertainment auctions. In the hotel auctions, WhiteDolphin bids quite differently compared to the other agents, placing an average bid of only 50 compared to Roxybot's 170 and Walverine's 115. Thus, we believe we could improve our strategy by bidding higher for the more expensive hotels and favouring more the hotel bonuses when deciding on which packages to bid for. Furthermore, we intend to force WhiteDolphin to deviate less from the clients' demands and thus incurring a lower penalty. This could be done by inflating the penalty for deviation from a customer's prefered dates to be much higher.

References

Bradley, D. M., and Gupta, R. C. 2002. On the distribution of the sum of n non-identically distributed uniform random variables. *Annals of the Institute of Statistical Mathematics* 54(3):689–700.

Cheng, S.; Leung, E.; Lochner, K. M.; O'Malley, K.; Reeves, D. M.; Schvartzman, L. J.; and Wellman, M. P. 2003. Walverine: A walrasian trading agent. *Proc. Second International Joint Conference on Autonomous Agents and Multi-Agent Systems* 465–472.

He, M., and Jennings, N. R. 2004. Designing a successful trading agent using fuzzy techniques. *IEEE Trans on Fuzzy Systems* 12 (3):389–410.

Lee, S. J.; Greenwald, A.; and Naroditskiy, V. 2007. Roxybot-06: An (SAA)^2 tac travel agent. In *Proc. of the Twentieth International Joint Conference on Artificial Intelligence*, 1378–1383.

Reeves, D. M.; Lochner, K. M.; Cheng, S.; and Suri, R. 2005. Approximate strategic reasoning through hierarchical reduction of large symmetric games. In *AAAI-05*.

Toulis, P.; Kehagias, D.; and Mitkas, P. 2006. Mertacor: A successful autonomous trading agent. In *Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, 1191–1198.

Vetsikas, I. A., and Selman, B. 2003. A principled study of the design tradeoffs for autonomous trading agents. *Proc. Second International Joint Conference on Autonomous Agents and Multiagent Systems* 473–480.

Vetsikas, I. A., and Selman, B. 2005. Autonomous trading agent design in the presence of tradeoffs. In *Proceedings of the 7th International Conference on Electronic Commerce, Xian, China*.

Vetsikas, I. A.; Jennings, N. R.; and Selman, B. 2007. Generating Bayes-Nash equilibria to design autonomous trading agents. In *Proc. of the Twentieth International Joint Conference on Artificial Intelligence*, 1543–1550.

Vytelingum, P.; Dash, R. K.; He, M.; and Jennings, N. R. 2005. A framework for designing strategies for trading agents. *Proc. IJCAI Workshop on Trading Agent Design and Analysis* 7–13.

Vytelingum, P. 2006. *The Structure and Behaviour of the Continuous Double Auction*. Ph.D. Dissertation, School of Electronics and Computer Science, University of Southampton.

Wellman, M. P.; Reeves, D. M.; Lochner, K. M.; and Suri, R. Searching for walverine 2005. In La-Poutré, H.; Sadeh, N.; and Janson, S., eds., *Agent-Mediated Electronic Commerce: Designing Trading Agents and Mechanisms (LNAI 3937)*.

Wellman, M. P.; Greenwald, A.; Stone, P.; and Wurman, P. R. 2002. The 2001 trading agent competition. *Proc. Fourteenth Conference on Innovative Applications of Artificial Intelligence* 935–941.