Bayesian Trading Agent

Calin Cenan
Computer Science Department
Technical University of Cluj-Napoca
Str. Baritiu Nr. 28, 400027
Romania
Calin.Cenan@cs.utcluj.ro

Abstract – **Software agents are increasingly being used to represent humans in on-line auctions. Such agents have the advantages of being able to systematically monitor a wide variety of auctions and then make rapid decisions about what bids to place in what auctions. To provide a means of evaluating and comparing research methods in this area the Trading Agent Competition (TAC) was established. This competition involves a number of agents bidding against one another in a number of related auctions, operating different protocols to purchase complementary and substitutable goods to form travel packages for a set of customers. Against this background, this paper describes a methodology for deciding the bidding strategy of agents participating in a significant number of simultaneous auctions, when finding an optimal solution in enough time is not possible. For the dimensions of TAC, an optimal solution to the allocation problem is not always tractable. Considering that the agent needs to solve an optimisation problem in order to maximise its gain we propose a genetic planner to determine the near optimal quantity of each resource to buy and sell given client preferences, current holdings, and market prices. In the presented experiments the genetic algorithms were used and in certain configuration of population size, number of evolution steps and mutation factor in almost all cases were obtained solutions which are higher then 95% of the optimal value. We continue with a description of ongoing and planned work about an adaptive and robust agent architecture based on Bayesian belief networks to solve the problems of desired prices and the time of bid placement. Using such adaptive uncertainty reasoning we hope to balance bid aggressiveness against the cost of obtaining increased flexibility.**

I. INTRODUCTION

Auctions are becoming an increasingly popular method for transacting business.

In practice agents are rarely interested in a single item, they wish to bid in several auctions in parallel for multiple interacting goods. In this case they must bid more intelligently in order to get exactly what they need.

For example a person may wish to make a PC from exactly one motherboard and one processor but if he does not have a flexible plan, he may only end up acquiring only one item. Goods are called complementary if the value of acquiring both together is higher than the sum of their individual values. On the other hand if that person bids for processors in several auctions, he may end up with more than one. Goods are called substitutable if the value of acquiring two of them is less than the sum of their individual values.

Researchers tested idea about agents in different market scenarios until Trading Agents Competition (TAC) was developed as a competing benchmark that incorporates several elements found in real market. Since it encapsulates most of the issues of the problems, it works as a appropriate universal test-bed for research ideas in the design of a trading agent. The TAC setting is designed to model a realistic market place setting, as might be encountered by, for example, in real marketplaces for realistic travel agents that organise trips for their clients. It includes several complementary and substitutable goods and a complex utility function.

In these setting instead on focusing only on bidding strategy, the computational problems are shifted also to agents who have to deal with complementary and substitutable goods and also compete against other agents.

II. MARKET GAME DESCRIPTION

A trading agent is a simulated travel agent whose task is to organise itineraries for a group of clients who wish to travel from a town to other and back again during a five day period of time. Travel and entertainment goods are traded at simultaneous on-line auctions that run in given interval of time (twelve minutes). There are obvious interdependencies, as the traveller needs a hotel for every night between arrival and departure of the flight, and can attend entertainment events only during that interval and only one event on a night. In addition, the clients have individual preferences over which days they are in destination town, the type of hotel, and which entertainments they want.

Each agent is acting on behalf of eight clients, who express their preferences for various aspects of the trip. An agent's objective is to secure the goods necessary to satisfy the particular desires of its clients, but to do so as inexpensively as possible. An agent's performance is measured by the difference between the utilities it earns for its clients and the agent's expenditures.

The market supply consists of three types of travel goods:

1. flights to destination and back
2. hotel room reservations at two different hotels; one hotel is all-around a nicer place to stay
3. entertainment tickets for three events in the destination town

There is a separate auction corresponding to every combination of good and day, yielding a total twenty-eight simultaneous auctions:

1. eight fight auctions (there are no inbound fights on the last day, and there are no outbound fights on the first day)
2. eight hotel auctions (two hotel types and four nights)
3. twelve entertainment ticket auctions (three entertainment event types and four nights)

All three types of goods (flights, hotels, and entertainment) are traded in separate markets with different rules.

1. an infinite supply of fights is sold at continuously clearing auctions in which prices follow a random walk
2. there are 16 hotel rooms per hotel per night available, which are sold at open-cry, ascending, multi-unit, sixteenth-price auctions
3. entertainment tickets are traded among agents in continuous double auctions; agents can act as either buyers or sellers, and transactions clear continuously

The market demand is determined by the forty-eight clients' preferences. Each client is characterised by a random set of six preferences for ideal arrival and departure dates, specific hotel room reservation value and reservation values for each of the three types of entertainment events.

For more details on the market game please visit http://www.sics.se/tac/.

III. AGENT ARCHITECTURE

The architecture of an agent, to be useful in solving general problems as issued by TAC, must be adaptive and flexible in order to make modifications and adjustments in the agent behaviour. Another reason for this requirement is that in most domains, and TAC is no exception to this rule, it is crucial to react fast to information on the domain and to other agents' actions. In addition, as information is gained by participation in the system, the agent architecture must allow and facilitate the incorporation of the knowledge obtained. Considering these we adopt a similar architecture with that proposed by Vetsikas and Selman in [8] and summarised in figure 1.

A. Price Estimates

In addition of the architecture we borrow from [8], after examining surveys on the TAC agents like [9] and [10] we started from the same "priceline" idea presented in [6] for implementing the first two modules.

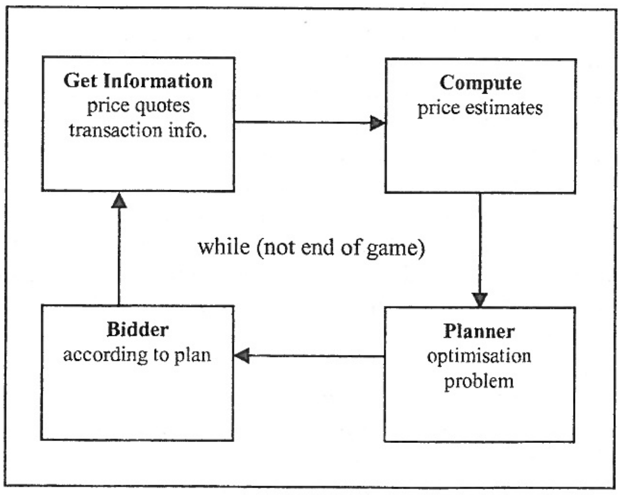

Figure 1. Trading agent architecture

B. Planner

We argue that bidding without an overall plan is obvious inefficient, because of the complementarities and substitutabilities between goods. So the planner is an important module in the agent's architecture. It uses the information and estimates obtained by the first two modules in order to determine the expense that the agent will probably have to pay for implementing the itineraries for its clients.

The agent must provide customers with complete itineraries and try to match them as much as possible to the customers' preferences in order to increase its income. So the planner must generate the customers' itineraries that maximise the agent's utility, which is the income from customers minus expenses. Therefore, it provides the type and total quantity of commodities that should be purchased or sold to achieve this.

This is an optimisation problem that the planner must formulate and solve. The problem is NP-complete, but for the size of the TAC problem an optimal solution usually can be produced in a reasonable time. In order to make the agent capable of a more rapid and flexible response instead of including elaborate heuristics we chose another approach. We consider that uncertainty is present in the definition of the problem because we cannot exactly estimate the others agents actions. Also at the beginning of the game the optimisation problem is based on inaccurate values, since the closing time of auctions are not known. Taking into account that a fully optimal solution is not necessary we argue that it can be replaced by an almost optimal approximation.

Considering all these and the observation that the problem the planner must solve it is an optimisation problem we came out with the idea of using a genetic planner as in [6] and [7].

C. Bidding Strategies

Once the plan has been generated the bidder places separate bids for all the commodities needed to implement the plan. In the way that the optimisation problem has been formulated, the solution only suggests what goods to buy, but not at what time, or at which price. To solve these issues we used empirical observations in order to build a number of Bayesian Belief Networks to act as rational element to solve these problems.

IV. GENETIC PLANNER

Planning is a difficult and fundamental problem of AI. An alternative solution to planning may lie in applying Genetic Programming to planning problems. We want to investigate the feasibility of using Genetic Programming to solve the problem of planning for a TAC agent. A Genetic Planning system was constructed based on information gathered from the previous modules.

Motivation for the main idea comes from supplying a new avenue for attacking the problem of planning. Genetic Algorithms, which Genetic Programming is based on, has often been used as an optimising tool with some considerable success in other areas of AI and it is hoped some of this success can be translated to the problem of planning for a TAC agent.

The possibility for applying GP to planning has already been demonstrated to work by [6] and [7]. Considering that to maximise the gain the agent needs to solve an optimisation problem in order to determine the optimal quantity of each resource to buy and sell given client preferences, current holdings, and market prices we propose to use genetic algorithms to solve the problem.

The Genetic Programming approach to planning is very different to the traditional approach of solving the planning problem. The classical work on planning is characterised by the planner reasoning logically about possible actions and how the actions relate in order to arrive at a correct plan. The Genetic Planner works by having a population of random plans and for each member of the plan population a fitness function assessed a fitness value. In order to compute the fitness function the plans must be simulated to determine their utility outcomes. Then the Genetic Planner proceeds by selecting the fit plans from the population for breeding and then recombining them to produce new individuals for the next generation of the population. This new individuals are hopefully more fit than the previous population. In this way the population is evolved to the correct solution.

The Genetic Planner algorithm starts with a process called seeding in which the original population of plans is constructed based on random values. The Genetic Planner algorithm starts by checking if some suitable terminating criterion has been reached. The termination criterion can be based on a minimum fitness value an individual has to reach or a prescribed maximum number of generations that can't be exceeded. If so the algorithm terminates with the best program as output.

Otherwise a new population is constructed from the old one. A fitness sensitive selection method, biased towards individuals with higher fitness, is used to select individuals form the population to have genetic operators applied to them. The selected individuals or parents are then used as input to the genetic operators. The genetic operators are used to create new plans for the next population. The two main genetic operators are reproduction and mutation. Reproduction takes a parent and duplicates it. A variant of this operator is crossover, which involves taking two parents and swapping a number of fragments of each parent between them in order to create two new members. The other genetic operator mutation involves slight alteration in the structure of a member from the population in order to force the optimisation process to search outside local maximum values in order to find a global maximum. These results are then added to population of the next generation and the process is repeated.

A simple method to do selection is tournament selection. Tournament selection works by pulling individuals with highest fitness to be selected as parents. In our approach the crossover is only one point and then combines the two parents to produce two offspring for the next generation. This process is done again to get other parents and other offspring. A parameter of our approach is the percent of population, which is obtained form crossover, usually larger, and the percent of next generation obtained from reproduction.

The main problem of the TAC planner is to solve the NP-complete optimisation problem of assigning purchased goods to clients and determining the bids, the optimal quantity of each resource to buy and sell given client preferences. Solving this problem we depart from approaches like in [1] and [2], of using domain-specific heuristic search, and we adapt a genetically solution.

In solving the planner problem for existing TAC agents, [9] and [10] there are two directions in which the solutions are split. A first direction is about looking for an exact or just a nearly optimal solution. The second direction is to solve to general optimisation problem for all eight clients of the traveller agent or to try to solve smaller problems of optimisation for only one client and then compose the solution for the original problem.

The solution of computing an exact value of the distribution of goods for all clients it is not very common between successful agents because this approach it is very complex and time consuming. A large majority of the TAC agents [9] and [10], which obtained good results in previous completions, used approximate approach to this planner problem looking only for nearly optimal solutions. Considering the existence of complementary and substitutable goods we argue that the overall utility cannot be exactly computed as a sum of particular utility of each client. Using Genetic Planner we will be able to obtain nearly optimal solution to the planner problem looking for an overall optimisation function for all the clients of the TAC agent. Moreover this Genetic Planner approach can be tailored to the exact requirements of time and performance of the agent.

In order to genetic code the trading agent we chose a representation compose for each individual of 104 genes. There are 8 clients and 13 genes for representing each client as follows:

- 4 bites to represent the 10 possible combinations of arrival and departure flights.
- 1 bit to represent the hotel type, either the fancier or the more convenient.
- 8 bites to represent the possible combinations for entertainment events; there are a maximum of 4 days available for attending entertainment and 4 possible events: no entertainment, and other three real possibilities.

For each individual the fitness is equal with the overall agent utility. The overall agent utility is computed in each situation based on client's preferences, already obtained goods and price estimations for all ongoing auctions. The actual configuration of the genetic coded trading agent gives an indication for the goods, which must be bought in order to satisfy the needs. After the already obtained goods are subtracted, based on data about auctions, estimation for the price that must be paid is computed.

In doing this we consider that bi-directional auctions exist and the agent can even sell some tickets to entertainment in order to obtain more utility that he can obtain by distributing them to his clients. We also consider that are possible closed auctions and a needed good at such auction is severely penalised.

The overall agent utility is finally the sum of each client's utility based on preferences from which is subtracted the sum paid for already obtained goods and the estimated price which must be paid for the rest of the needed goods.

V. BAYESIAN BELIEF NETWORKS

The presence of other agents in trading complementary and substitutable goods at multiple indeterminate auctions makes TAC a very uncertain domain. Unlike [4] we choose to deal with this uncertainty using Bayesian belief networks as a tool for modelling the decision-making and to reason under uncertainty.

This knowledge representation method use a special type of diagram called a direct acyclic graph together with an associated set of probability tables. The nodes in such a graph represent variables, which can be discrete or continuous and the arcs represent causal relationships between variables. The values on the arcs are actually the conditional probability of the child variable given the variable parent for which we have to provide a probability for each combination of events. In a nutshell, these are graphical representations of the probabilistic relationships between a set of variables.

Bayesian belief networks provide a way of encoding known correlation among a set of variables, such as the conditional independence relationships. With this knowledge representation method it is possible to articulate expert beliefs about the dependencies between different variables and to obtain a scientific rigour when the probability distributions are constructed. Also the ability to graphically depict causality makes them much easier to interpret then other AI techniques.

There may be several ways of determining the probabilities of any of the tables. For example we might be able to base the probabilities on previously observed frequencies occurring in practice or alternatively, if no such statistical data is available, we may have to rely on subjective probabilities entered by experts. In addition, more advanced learning techniques can actually determine a proper structure for the network and expose causal relationships that were previously unknown.

The key feature of Bayesian belief networks is that they enable us to model and reason about uncertainty when we apply the probability theory to propagate consistently the impact of evidence on the probabilities of uncertain outcomes. Basically the evidences are observations given to us and our task is to calculate the posterior marginal probabilities, or expectation, for a subset of variables.

In our approach we will use a software package called JavaBayes in order to implement this process. In order to perform inference, we will set our evidence by taking the variables we know and set them to the proper values. Then we will evaluate the network and read the new probabilities for each node in which we are interested.

Post competition analysis as [9] and [10] shows that agent's performance depends on a number of elements as the time and the type of environment in which competes. Based on these we choose to divide the game into three stages: probing stage (up to minute 4), decisive stage (from minutes 5 to 10) and the rest is finalisation stage. Also we set up our Bayesian belief networks based on three types of environment for two types of agents:

- *Risk-averse* agent is one that buys goods at the beginning of the game and that bids for complementary goods as the game progresses. These agents are highly flexible and cope well on competitive environments.
- *Risk-seeking* agent buys a large number of goods at the beginning of the game and seldom changes the travel plan of its customers during the game. This kind of agent does well in non-competitive environments in which goods are cheap

The three types of environment [3] and [5] are:

- *Competitive environments* where the prices of the goods are high. This is caused either by some agents insisting on bidding for hotels even when their ask price becomes high or the fact that some agents increase bids sharply rather than gradually.
- *Non-competitive environments* where there is very little competition for goods and an agent can obtain the products it wants at low prices.
- *Semi-competitive environments* where prices are medium. There is competition, but it is not severe.

A. Flight auctions

In the first probing stage the agent submits fixed low bids for all available flights to take advantage of possible low airfares. In the last finalisation stage the agent assures the booking of the flights that match the anchor solution for each client by submitting even high bids.

B. Hotel auctions

Hotel auctions are the most important and difficult part in Trading Agent Competition. According to the basic laws of microeconomics, the higher demand there is in a market, the higher the price of the goods. Hotel auctions are also the most uncertain part of the game. This uncertainty stems both from the random nature of the customers' preferences and from the way opponents deal with their hotel bidding.

Nevertheless, a rational agent should have submitted bids for hotels during the probing stage. In this stage the demand on the hotel market is unpredictable. Given this, we will have a rationale to buy some flights, with a high probability of needing. As the decisive stage progresses, the demand of the various auctions becomes clearer and rooms are actually allocated which means the agent can more accurately decide bids to go for. The finalisation stage represents the agent's last chance to transact on entertainment tickets and to buy any remaining flights and hotels that are needed.

The Bayesian belief networks used by hotel bidder reflect that the hotels in greatest demand are for second and third day and the nicest is favoured. Also, the rooms of the second day have a close relationship with those of third day because many customers stay for successive days. Others factors that affect the bid of hotels are the ask price of that auction, the counterpart hotel ask price and if applicable the closing time, the current stage into the game and the rate of change of the hotel ask price.

VI. DISCUSIONS, RESULTS

We will present the results of our genetically experiments and we will recommend a configuration to work on. We will conclude that using such a genetic planner can solve the allocation problem in the majority of cases with near optimal results.

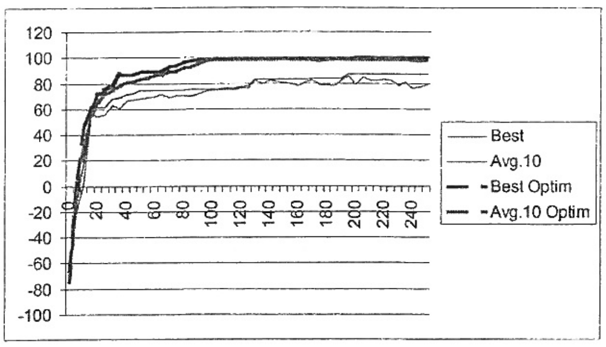

Figure 2. Solutions obtained in two genetic experiments, best utility and an average value for the first 10 individuals from a genetic population of trading agents are presented.

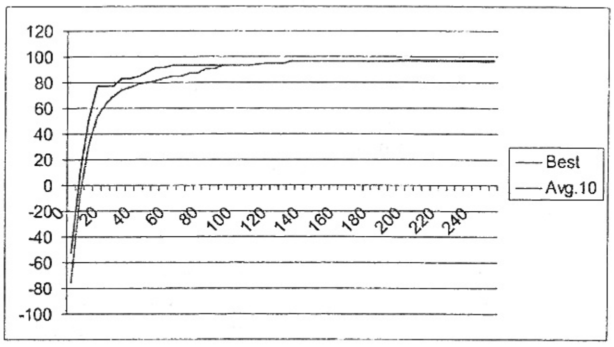

Figure 3. In recommended configuration, best utility and an average value for the first 10 individuals from a genetic population of trading agents.

Overall agent utility, the function to optimise, is a very difficult one. The reasons is that are a lot of large jumps caused by the fact that achieving the exact preferences for arrival, departure and hotel type for a client induce an increase in the utility of the agent of hundred units. Also in the function are a lot of small jumps caused by the fact that there are differences of only units between giving entertainment tickets to different clients in different days.

We limit our genetic experiments to four minutes and we use a fairly common computer configuration (AMD Athlon XP processor at a 2.5 GHz frequency). Using small population of trading agents and large number of evolution cycles we did not obtained concluding results. Using very large populations we obtained conflicting results. Quite often (5-10%) we obtained optimal solution but in the rest of the cases we obtained only poor results, 80-90% of the best utility.

In the previous figure can be seen in a continuous line a typical case for such an experiment, which is blocked at less than 90% of the optimal solution, and in a dotted line a very rare case that arrive at the optimal solution. For both the two agents are presented in the figure the utility for the best genetic agent from the population and the average value for the first 10 individuals.

We recommend using around 5000 trading agents in the population for around 500 evolution cycles. We also recommend using high crossover factor and mutation factor for inducing a rapid convergence and respectively to force out the optimisation function from the numerous local maxims. In such conditions we obtained very good solutions on the problem of finding the best configuration for a trading agents to maximise the utility function. It is true that we obtained very infrequent the optimal solution, only in 2 or 3 times, but in the majority of cases we attained more then 95% of the optimal value.

TABLE 1
TYPICAL RESULTS OBTAINED

Secs.	*Best*	*Avg.*	*Secs*	*Best*	*Avg.*
0	-52.53%	-75.92%	*150*	96.81%	96.80%
30	83.12%	73.97%	*180*	96.95%	96.62%
60	93.61%	84.84%	*210*	97.65%	96.99%
90	93.61%	93.56%	*240*	97.65%	96.49%
120	94.81%	94.80%	*250*	97.65%	96.95%

From more then one hundred experiments in different situations, in less than 10% we obtained between 90% and 95% form the optimal utility and in more then 90% of the cases we obtained between 95% and 99% from the optimal utility. In table 1 are presented the average results obtained after 100 such genetic experiments in different cases. In figure 3 a graphical representation of these data is also presented.

VII. CONCLUSIONS

To verify our observations and show the generality of our conclusions we run experiments in the given conditions for more evolution cycles and more then 4-5 minutes and we obtained similar results. Even changing the mutation factor at the same population size we did not obtained increasing in the overall utility and the fitness in more evolution steps. We can conclude then, that the behaviour of the optimisation function is dictated by the convergence properties of the genetic algorithms and in the recommended configuration can be used as a planner part for a TAC agent.

In this paper we also proposed architecture for bidding strategically in simultaneous auctions. We argue that dealing with the inherent uncertainty the Bayesian belief networks will prove a great flexibility of the agent. We also argue that in such architecture the Bayesian belief reasoning will stick very well with the genetic planner. The genetic planner will add flexibility and power solving anytime during the trading, in short time, the optimisation problem of determining together both the best distribution for the acquired goods and the best bid in order to maximise the TAC agent's utility.

VIII. ACKNOWLEDGEMENTS

Part of this work was supported by Romanian CNCSIS under Grant 528/2002.

VIII. REFERENCES

[1] L. Ding, T. Finin, Y. Shi, Y.Zou, Z. Ding and R. Pan "Strategies and heuristics used by the UMBCTAC agent in the 3rd Trading Agent Competition", in *Proc. Workshop Trading Agent Design and Analysis*, 2003.

[2] A.R. Greenwald and J. Boyan "Bid determination for simultaneous auctions", in *Proc of the 3rd ACM Conference on Electronic Commerce*, 2001, pp. 115-124, 210--212.

[3] M. He and N.R. Jennings, "SouthamptonTAC: An adaptive autonomous trading agent" *ACM Trans on Internet Technology*, vol. 3 (3), 2003, pp.218-35.

[4] M. He, H. Leung and N. R. Jennings, "A fuzzy logic based bidding strategy for autonomous agents in continuous double auctions" *IEEE Trans. on Knowledge and Data Engineering*, Vol. 15 (6), 2003, pp.1345-63.

[5] M. He and N. R. Jennings "SouthamptonTAC: Designing a successful trading agent" in *Proc. 15th European Conf. on AI (ECAI-2002)*, Lyon, France, pp. 8-12.

[6] I. Muslea "A General-Purpose AI Planning System Based on the Genetic Programming Paradigm ", in *Late Breaking Papers at the Genetic Programming 1997 Conference*, 1997.

[7] L. Spector "Genetic Programming and AI Planning Systems", in *Proc of the Twelfth National Conference on Artificial Intelligence, AAAI-94*, 1994, pp. 1329-1334.

[8] I.A. Vetsikas, B. Selman, "A principled study of the design tradeoffs for autonomous trading agents", in *Proc. AAMAS*, 2003.

[9] M.P. Wellman, S.F. Cheng, D.M. Reeves and K.M. Lochner "Trading Agents Competing: Performance, Progress, and Market Effectiveness" *IEEE Intelligent Systems*, vol.18(6), 2003, pp. 48-53.

[10] M.P. Wellman, A. Greenwald, P. Stone, and P.R. Wurman "The 2001 Trading Agent Competition" *Electronic Markets*, Routledge, vol. 13(1), 2003, pp. 4-12.