Research on Modeling Other Agents

Referent:
Björn Westman

Based on:
Towards Modeling Other Agents : A Simulation-Based Study
By: Leonardo Garrido, Ramón Brena and Katia Sycara

Dozenten:
Michael Schillo
Christof Klein

Year: 2000

TABLE OF CONTENTS

Introduction **2**

The goals and the methods **2**

Agents modeling and related literature **3**

The Meeting Scheduling Game **4**

Strategies **6**
Random Strategy 7
Oracle Strategy 7
Team-Size Strategy 8
Preference Strategy 8
Semi-Oracle Strategy 8

Experimental results **9**

Conclusions and reflections **12**

References **13**

Introduction

In multiagent systems, agents beliefs about other agents are specially crucial since the other agents are in many cases the most important part of the agent's environment. The paper "Towards Modeling Other Agents : A Simulation-Based Study" written by Leonardo Garrido, Ramón Brena and Katia Sykara[1] is basically about investigating the advantages one can obtain by modeling other agents in multiagent environments. Garrido et al have tried to do this by quantifying the value of building models about other agents using no more than observation of others' behaviour.

In this paper I will describe the goals and the methods used by Garrido et al. The first section is about the goals of the study. It also briefly describes what method the authors of the paper have used to obtain their goals. The second part deals with the question of what agents' modeling is and refers to some related literature on the subject. In the third part I will go more in to detail on how the experiments have been conducted. Garrido et al have used a testbed where competition takes place in a game that has characteristics of meeting scheduling problems. I will describe the framework of this game, called "The Meeting Scheduling Game" (MSG) in detail.

In the section about the MSG you can read about the different roles that are defined for the agents in the experimental framework. Moreover you can read about different agent strategies. This part of the paper is crucial since it explores a range of agent strategies, from "blind" randomized strategies to "oracle" strategies where the agent has all the necessary information about the other agents. The experiments conducted and the points made by Garrido et al are built up around the different agent strategies.

The goals and the methods

The main goal of the study is to research the competitive advantages an agent can obtain by modeling some important aspects of its competitors. In the paper, Garrido et al want to show that a *modeler agent* can take advantage of building and updating beliefs about other agents. They also want to show that this advantage can make it perform better than agents without modeling capabilities.

Another interesting aspect of the research is that Garrido et al intended to study modeling other agents using no more than other agents' behaviour.

The method used to achieve the goals is a simulation based study. In the testbed competition takes place in a game where meeting scheduling problems arise. It is basically about a group of people who are trying to arrange a meeting in such a way that a certain meeting slot is available

for as many group members as possible. Furthermore the meeting slot should be as convenient as possible for everyone. This means the fulfilment of individual preferences should be maximized. The game is called "The Meeting Scheduling Game" and will be described later in this paper.

In the framework of MSG, Garrido et al explores -in an experimental way- how modeling other agents can affect individual and group performance after a series of meetings. They want to allow both self-interest as well as co-operative behaviour. Garrido et al describes how their basic research-driving hypothesis is that both individual and group performance are better when each agent tries to explain the other agents' behaviour in terms of internal cognitive structures or models - creating and adapting these agent models in an incremental way.

Agents modeling and related literature

Research on modeling other agents has been approached from different perspectives. Garrido et al presents work related to theirs mainly concerning three major issues regarding agents' modeling:

- What is to be modeled?
- How is it to be modeled?
- How are models built?

There is a big range of agent models. From very specialized , task-oriented models to structural models using some kind of internal structure of the agent being modeled. Structural models are often referred to as "cognitive" or "deep" models while the specialized models are often called "surface" models. Deep and surface models can also be combined. According to Garrido et al one of the advantages of deep models is that they allow predictions of others' behaviour, possibly resulting in an advantage in non-co-operative or non-communicating settings.

According to Carmel and Markovitch [2], it is important to model others' strategies in order to perform better against them in a competitive setting. Carmel and Markovitch have presented an heuristic algorithm to infer a model of the opponent's strategy, represented as a Deterministic Finite Automaton (DFA), in their approach to modeling other agents. Another interesting work on opponent modeling has been presented by Sen and Arora [3]. They propose a scheme for learning opponent action probabilities and a maximum of expected utility strategy for exploiting weaker opponents.

When talking about the model construction method, the probably most simple case arises in co-operative settings, where honest agents "tell" the others what their characteristics are [4]. This does not work in a competitive setting of course. A very complex situation arises when

others' models are built entirely from the observation of others' behaviour.

In their latest research Garrido et al say they view agent modeling as an iterative and gradual process, where every new piece of information about an agent is analysed in such a way that the model of the agent is further refined, using a bayesian updating mechanism. There have been other papers sharing this view, for instance: Gmytrasiewicz at al[5] have proposed a framework for bayesian updating of agent models within the formalism of RMM.

Bayesian modeler agent
A bayesian-modeler agent builds models about the others in an incremental and iterative way, updating those models after each round during the whole game. All the probabilities of each model are incrementally updated, trying to reach the actual character of the agent being modeled.

In the next section I will describe the experimental framework of the research, the Meeting Scheduling Game and describe the different agent roles and strategies used in the study.

The Meeting Scheduling Game

In this section I will give a brief description of the Meeting Scheduling Game (MSG). The main idea of the game is as follows: The players try to arrange a meeting at some convenient time-slot, convenient referring to a slot that has an acceptable *utility* for a specific player, according to his *preference profile* or *utility function* . Each player has a role which is defined by the preference profile. The preference profile is coded as a calendar slot utility function, ranking each slot from the most preferable to the least preferable one. In MSG Garrido et al have defined a number of agent roles. The roles described below are some of the most basic and familiar ones:

- **The Early-Rising.** An agent with this role prefers the early hours of the day.
- **The Night-Owl.** An agent with this role prefers it when the meetings are scheduled as late as possible.
- **The Medium.** An agent with this role prefers the meetings to be around noon.
- **The Extreme.** An agent with this role prefers to have meetings scheduled either early in the morning or late in the afternoon.

Figure 1. shows examples of these roles with four arbitrary eight-slot calendars.

The night owl (busy slots 1, 3, 5, 6)

Utility	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
	-		-		-	-		
Time slots	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**

The medium (busy slots 1, 2, 3, 4)

Utility	*1*	*3*	*5*	*7*	*8*	*6*	*4*	*2*
	-	-	-	-				
Time slots	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**

The early-rising (busy slots 2, 3, 7, 8)

Utility	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*
		-	-				-	-
Time slots	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**

The extreme (busy slot 1)

Utility	*8*	*6*	*4*	*2*	*1*	*3*	*5*	*7*
	-							
Time slots	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**

Figure 1. Four basic agent roles with eight-slots calendars.

One of the main concerns that the developers of MSG had when creating the testbed was to allow self-interest as well as co-operative behaviour. They wanted the players to be able to show and/or hide personal information and they wanted it to be possible to define different player's strategies in addition to the roles. The combination of role and strategy defines a player's preferences and behaviour and the conjunction of role/strategy is therefor refereed to as a player's *personality*. As in any other competitive game, the goal of a player is to accumulate more points than his opponents in a match. One *match* consists of a fixed number of *games* (e. g. ten) and each *game* consists of two *rounds*.

In the first round of each game, each player bids for the slot which maximizes his utility, according basically to his own role. However the player bids are not completely determined by the own players role since the calendars are pre-set with something called calendar density. This means that some slots, for example the most preferable may already be busy in the player's calendar. In this case the player chooses the most preferable of his available slots. All the information about each player's bid is *public knowledge* announced by the *referee*. The referee is also an agent. He ensures that all the players obey the rules of the game. He is also responsible for accumulating points for each agent after each game in an individual point counter for each player during the whole match.

When the referee has announced the bids from the first round he calls for the second round. The second round is pretty similar o the first one.

However there is a significant difference, each player can now follow different strategies, taking into account the first-round information of the current game, historic records of past games, and/or models of the other agents. After all players have made their second round proposal, a number of *teams* arise. Each team is composed of all those players who proposed the same calendar slot in the first round. Then the team joint utility is calculated, summing up all the team members´ calendar utilities:

$$TJU(t) = \sum_{\forall m \in t} U_m(s_t)$$

Here, t is a team, m is a member of the team , s_t is the slot proposed by the members of t, U_m is the slot utility of member m. Finally the game is won by the team which accumulates the greatest team joint utility.

Once the winning team is selected , each agent earns points according to the following predefined *scoring procedure*: all the players outside the winning team accumualte zero points for that game and each agent *a* in the winning team *t* accumulates his own slot utility plus the team joint utility: $TJU(t) + U_a(s_t)$. The purpose of this mixed procedure is to promote a balance between selfish and collaborative attitudes. Finally, the winner of the match is the player with the most individual points accumulated at the end of the last game.

After a game, each player's calendar is randomly reset with at the predefined calendar density and another game is started. All this process is repeated until the predefined number of games have been played.

It is worthy to note that, after each round , each player knows only the other agents´ bids and, after each game, he knows only his own accumulated utility. The referee has the responsibility of keeping all other information, for example role and strategy, private.

Strategies

Besides its role, each agent also have a particular game *strategy*. Strategies are rules that describe and tell agents how to act at some specific decision point -in MSG this is at the second round of each game. A modeler agent is nothing more than an agent using a strategy which builds models about other agents, and uses these models in order to improve its behaviour.

In order to explore the whole spectrum of strategies ranging from the least- to the most-informed one, the researchers first defined the following two strategies:

Random Strategy

An agent using the random strategy chooses his next bid among its action set using a uniform probability distribution. This means he all possible actions have the same probability.

A random agent does not take into account any information about the other agents nor does it consider its own agent role. The only thing it takes into account is its own calendar since the slot it proposes must be an available one.

The Random Strategy's performance is used as a reference point to compare all the other strategies. Any strategy performing worse than the random strategy is neglected and considered unreasonable.

Oracle Strategy

An agent using this strategy can see in advance the others´ next move because he indeed knows the other agents´ calendars, roles and strategies. For each free slot s in his calendar, he calculates his possible gain $G_o(s)$, if he proposed that slot. Then he finds the agent m who would earn the maximum gain $G_m(s)$ among the rest of the players, if he proposed that slot. Then he calculates the utility of each slot s as his gain with respect to the profit of agent m:

$$\mathbf{U}(s) = G_o(s) - G_m(s)$$

After checking all his free slots, he proposes the slot with the highest utility: $\arg\max_s \mathbf{U}(s)$.

In the paper, the definition of an oracle agent states that an agent with the oracle strategy is able to correctly guessing the other agents´ roles and strategies, in the actual implementation the referee agent gives the oracle agent the other agents´ information (roles, strategies and calendars). That's why the oracle agent can always see in advance the others´ second-round bid and calculate all the possible gains he could get under all different playing options, also taking into account his own calendar availability.

With the Random Strategy being the lower limit of an agents performance the Oracle Strategy is the upper limit. One might think that an oracle agent will always win every match. This is however not the case, the oracle agent can not always win! This is because of the calendar density. An oracle agent can not choose the best possible slot every time since it might not be available.

The next two strategy were also quite simple, a little more complex than the random strategy though:

Team-Size Strategy

An agent using this strategy has the goal of always joining the biggest team seen after the first round with his second bid, if this slot is available. As with the random strategy, this one does not take into account any information about other agents´ roles.

Preference Strategy

This strategy tries to maximize the agent's utility based on his preference profile. This comes to simply choose the same slot in the first and the second round, namely the one with the highest utility. The preference strategy does not take into account any information about the other agents.

In order to get a more refined upper limit than the oracle strategy, they then defined a more realistic oracle strategy.

Semi-Oracle Strategy

The Semi-Oracle Strategy is similar to the Oracle-Strategy. It guesses information about the others´ roles and strategies; however it does not take into account the other agent's calendars.

A semi-oracle calculates the most probable bids of the other agents. In order to do this it takes into account the calendar density. The main idea with this agent is to maximize the *expected utility*, which is the product of a utility and the probability of getting it. The general behaviour of a semi-oracle agent is as follows:

1. For each other agent a, make a set S_a of all possible slots that can be proposed for agent a.
2. For each possible slot s_a in each set S_a, calculate the probability $P(s_a)$ of being actually selected by agent a.
3. Combining all the possible slots that each agent can propose, generate the set O of all the possible outcomes or playing scenarios that could arise in the second round. Each possible outcome o in O is a vector o def= (s_1, ..., s_n) of slots that can be proposed by each other agent in the second round.
4.For each possible outcome $o \in O$ do:

- **4.1 Calculate the probability $P(o)$ that o arises at the second round.**
- **4.2 Find which would be the slot s_o that gives the maximum utility u_o that can be earned by the semi-oracle agent in this outcome o.**
- **4.3 Calculate the semi-oracle's expected utility $U(s_o)$ due to slot so under this outcome o.**
- **4.4 Accumulate $U(s_o)$ to the previous expected utilities due to the same slot s_o obtained in the previous outcomes.**

5. Choose the slot s_m with the maximum accumulated utility.
6. Bid for the slot s_m.

Now, let us look more in detail on step one and two: it is worthy to note that all the possible slots that can be proposed by any preference agent is always the same slot that was proposed in the first round of bids. The probability of this is obviously 1; however the set of slots that can be proposed by any team-size agent is composed of the different slots proposed by all the other agents.

In order to calculate the probability of each possible slot, the semi-oracle agent must perform the following steps:

1. See what teams arise in the first round of the game
2. Make sets of teams according to the team size (make sets of teams with the same number of members).
3. Make a vector v = (e_1, ..., e_n) with all the sets of teams made previously. Arrange this vector in descendant order, according to the size of the teams of each set, from the set e_1 with the biggest teams to the set of teams en which contains the team that contains the team-size agent *a*.
4. For each set of teams $e_i \in$ v, calculate the probability of being chosen by the team-size agent, given the known calendar density d.
5. Calculate the conditional probability $P(t_a | e_n)$ of choosing team $t_a \in e_n$ which contains the team-size agent a given that e_n is already chosen.
6. Now for each team $t_j \in e_i$ of each set of teams e_i, calculate the probability $P(t_j)$ of being chosen for joining this team. At this stage three different situations can arise: The first case is when the team-size agent *a* is not in any team t_m of the set e_i, the second case is when the team-size agent *a* is not member of the team t_j but it is a member of some team other team $t_m \in e_i$; and the third case is when the team-size agent *a* is member of the team t_j and it is in the set e_i.
7. Assign the probability $P(t_j)$ to the slot proposed by t_j, this slot is one of the slots s_a that each team-size agent a could propose in step 2 of the previous algorithm.

The semi-oracle agent might appear very similar to the oracle agent, however it is much more complex. While the oracle agent can predict the others´ moves in an exact way, the semi-oracle must calculate the probabilities of all the possible bids of the other agents.

Garrido et al concludes the section about strategies by saying that the oracle strategy provides an upper limit performance for any modeler agent. Furthermore, the semi-oracle strategy can give us a smaller, more refined upper limit. In the following section, about the results of two different experimental scenarios we will see how these upper limits can be empirically obtained.

Experimental results

In the experimental scenarios matches of ten two-round games were set up. A series of matches were set up in order to measure, after many matches, how agent performance was affected by different strategies. Each experiment conducted in the study was composed of 500

independent matches. Once a match is completed it is referred to as a success if the strategy currently considered wins. Otherwise it is referred to as a failure. assuming that the success probability p remains constant through all the matches one can draw the conclusion that the experiments are binomial.

In all the experiments the mixed scoring procedure described earlier has been used. The calendar density was set to 50%. This density was chosen with respect to other experiments, not presented in the paper, which show that a lower density lead to scenarios of little interest and high densities lead to chaotic scenarios because agent almost never can play their strategies- they show a kind of pseudo-random behaviour.

The goal of the first experimental scenario, the random, the team-size and the preference strategy was playing against each other. First random agents were to play against the team-size agents. In the second experiment, the researchers ran random agents against preference agents. In the third experiment the team-size and the preference agents played against each other. The results were as follows:

Experimental Scenario 1

	Strategies		
Experiments	*Random*	*Team-Size*	*Preference*
1.1	8.04% (5%error)	91.96% (5%error)	-
1.2	0.60% (1%error)	-	99.40% (1%error)
1.3	-	50.30% (6%error)	49.70% (6%error)

This first experiment in this scenario clearly shows that the team-size strategy with ease outperforms the random strategy. The second experiment shows that also the preference strategy outperforms the random strategy. In the last experiment in this scenario, the team-size strategy and the preference strategy are played against each-other, from the result in this experiment it is not clear which strategy is better. Therefor Garrido et al have conducted another set of experiments not detailed in the paper which shows that the team-size strategy has better performance if the number of agents are greater than a specific threshold. Garrido et al explains this by stating that team-size agents tend to gather together in the same team. After a specific number of running agents, the team-size team outperforms any other team.

The next table presents the results of the second experimental scenario.

Experimental Scenario 2

Strategies				
Experiments	*Oracle*	*Semi-Oracle*	*Team-size*	*Preference*
2.1	59.22% (7%error)	-	19.57% (7%error)	21.32% (7%error)
2.2	-	53.40% (7%error)	23.76% (7%error)	22.84% (7%error)

Experiment 2.1 shows the upper limit as obtained by the oracle strategy. The oracle strategy clearly outperforms the two other strategies. Experiment 2.2 clearly shows that the semi-oracle strategy outperforms the team-size and the preference strategy, as expected after seeing the results of the first experiment in this scenario.

The gap between the two oracle strategies and the two more basic strategies is relatively big. Garrido et al draws the conclusion that anywhere in this gap any modeler agent can be located. Since the performance of the semi-oracle strategy is relatively close to the oracle strategy performance the results of experiment 2.2 does not contradict the conclusion made.

Conclusions and reflections

In the study, the experimental framework is described as one which makes it possible to assess the relative performance of different plying strategies for the MSG. According to Garrido et al this allows them to evaluate the performance strategies that build and use models of their competitors. According to Garrido et al other modeling-agents proposals fail to provide this kind of evaluation.

One thing that I note when I read the paper is how Garrido et al talks about modeling other agents almost as if it was an easy procedure. At least you get the impression that oracle- and semi oracle agents are not so hard to implement. However Garrido et al does not perform this implementation, they simply let the referee provide the modeler agents with the information about the other agents. Of course their ongoing research might provide us with a more refined picture of the use of modeler agents. As the authors say in the paper the result of this study is not much surprising, this obviously does not mean that the research is useless but I ask myself how groundbreaking and interesting the study is at this point.

The study does make use of a collection of reference points for a modeler agent's performance: The random agent and the oracle provide the extremes of the spectrum, ranging from least-informed to most-informed strategies. Garrido et al also says that they, in their ongoing research has formally defined something called a pre-modeler agent, which is not an oracle agent, but it is capable of using a pre-built probabilistic model.

It can also be said that the strategies used in this study does not really cover the whole spectrum. The basic strategies are pretty similar to one another and so are the modeling strategies. It would be nice to see how a agent using a strategy even more in-between the basic strategies and the oracle strategies would perform. As it is now, the semi-oracle strategy has all the information he needs to make the best possible choice accept for the calendar information of the others. I think it would be interesting to see how a modeling agent which has perhaps not all the information but say 75% of the information right would perform in this setting. The reason why I find this interesting is because surely a modeler agent only using the observation of others´ behaviour can not possibly guess the strategies and the roles of the others´ correctly every time. Sometimes it has to make an incorrect assumption. At least at this point in the development of modeling agents.

This study wanted to provide a testbed where selfish as well as co-operative behaviour is promoted. However the goal for each agent was to win the game. Only one could be the winner. The points gained from the Team joint utility was far greater than the points gained from the individual utility. This promotes teaming up with others rather than considering the agent's own preference profile. However in experiment

1.3 in this paper one can see that the team-size strategy and the preference strategy are pretty equal when played against each other. Then the authors state that the team-size strategy always wins when a certain number of agents take part. They do not provide this number and nor do they say how many agents where in the game in the experimental scenario. I find it a bit peculiar and disturbing that this kind of vital information about the experimental framework is left out.

Finally I would like to say that I found the paper interesting and I find the approach good. I have read some of the more resent research made on the subject and I think this study and the paper that it resulted in provided a good starting point which further research can build on. I said earlier that perhaps the results of the study are not all that surprising but nonetheless it is research that has to be made in order to see that there is a reason to continue researching on modeling agents in competitive settings.

References

[1] L. Garrido, R. Brena, and K. Sykara. Towards modeling other agents: A simulation based study. Multi-Agent Systems and Agent-Based Simulation LNAI Series, Volume 1534. Springer-Verlag. Berlin, December, 1998.

[2] D. Carmel and S. Markovitch. Opponent modelling in a multi-agent systems. In G. Weiss and S. Sen, editors, Lecture notes in AI, 1042: Adaption and learning in multi-agent systems, Lecture notes in Artificial Intelligence. Springer-Verlag, 1996.

[3] S. Sen and N. Arora. Learning to take risks. In AAAI-97 Workshop on Multiagent Learning, 1997.

[4] R. Davis and R. G. Smith. Negotiation as a metaphor for distributed problem solving. Artificial Intelligence, pages 63-109, 1983.

[5] P.J. Gmytrasiewicz, S. Noh, and T. Kellogg. Bayesian update of recursive agent models. Journal of User Modeling and User-Adapted Interaction, 8(1/2):49-69, 1998.