Towards Social Complexity Reduction in Multiagent Learning: the ADHOC Approach

Michael Rovatsos and **Marco Wolf**
{rovatsos,wolf}@cs.tum.edu
Institut für Informatik
Technische Universität München
80290 München, Germany

Abstract

This paper presents a novel method for classifying adversaries that is designed to achieve *social complexity reduction* in large-scale, open multiagent systems. In contrast to previous work on opponent modelling, we seek to generalise from individuals and to identify suitable *opponent classes*. To validate the adequacy of our approach, we present initial experiments in a multiagent Iterated Prisoner's Dilemma scenario and we discuss directions for future work on the subject.

Keywords: Opponent modelling, model-based learning, social learning, Iterated Prisoner's Dilemma.

Introduction

In artificial societies that consist of self-motivated, possibly antagonistic and non-benevolent agents, it has long been recognised that modelling the "other" agent is essential to achieving fruitful coordination, if no *a priori* reliable information is available to predict others' strategies. Very often, *learning* opponent models (Vidal & Durfee 1997; Carmel & Markovitch 1996a; Bui, Kieronska, & Venkatesh 1996; Balch 1997; Rovatsos & Lind 2000; Stone 2000) may be the only way to derive useful information about others' future actions by acquiring a model of their behaviour, by reconstructing their internal reasoning mechanism and, possibly, by deliberately "massaging" them into their most cooperative stance (Freund *et al.* 1995).

However, in large-scale MAS in which agents have only occasional encounters with peers they are acquainted with, learning models of *individual* opponents may be inefficient, because the cost of acquiring and maintaining an adequate model of the other may outweigh its potential benefits if the probability of interacting with that same agent in the future is not very high. Therefore, it seems inappropriate to discriminate between each and every peer agent.

One way to overcome this problem is to lay a greater focus on behaviour repeatedly observed *across* several peers than on that of the individual, e.g. by restricting the number of potential "models of others" to a relatively small number of opponent behaviour "types" or "classes". Particular agents are then mapped to these classes according to the behaviour they exhibit, and optimal strategies of how to act toward other agents are learned with respect to agent types rather than agents.

By means of such *social complexity reduction* (which is a common phenomenon in human societies, if we only think about stereotypes, prejudice against social groups, etc.) learning agents exploit regularities which quite naturally exist across agents in a MAS, and which may stem from different "configurations" of agent motivations, of types of access to resources, etc. Besides the advantages that this approach has with respect to computational cost (compared to that of granting each peer its own model), it also has the capacity to speed up the individual opponent modelling subprocesses, because these are provided with more "training data-per-model", as they make use of encounters with more than one agent.

In this paper, we present the ADaptive Heuristic for Opponent Classification ADHOC which evolves k models of "peer classes" in settings in which the modelling agent interacts with n peer agents, where, typically, $n \gg k$.

The number of necessary classes is computed *on-line* during the learning and interaction process as well as the agent-class membership function, and both are subject to constant revision. For each class, the agent learns an optimal strategy using some opponent modelling method (OMM)[1] and employs it in encounters with members of that class.

Experiments in multiagent Iterated Prisoner's Dilemma (IPD) (Axelrod 1984) simulations demonstrate that ADHOC agents succeed in learning arbitrary sets of given opponent strategies and in correctly classifying their peers, and that their overall performance during the game simulations lies above that of "unboundedly rational learning agents" which are capable of constructing one model per adversary.

Since the heuristic does not rely on any particular choice of OMM, we have reason to believe that it carries over to more complex applications and that its elaboration deserves further study. More generally, we believe that the principle of *social complexity reduction* that it realises is crucial to overcome the mentalistic AI tradition (that is very much concerned with optimally modelling the mental processes of

[1]The OMM is not part of the heuristic itself; here, we use a combination of Carmel and Markovitch' (Carmel & Markovitch 1996b; 1996a) method for learning deterministic finite automata (DFA) and Q-learning (Watkins & Dayan 1992) as described below.

individuals) in order to build highly adaptive MAS with a particular focus on scalability and bounded rationality (Russell & Wefald 1991).

The remainder of this paper is structured as follows: in the next section, we introduce the heuristic and explain the intuition behind it as well as the underlying assumptions. Then, we describe the simulation scenario and the particular OMM used in our experiments. The subsequent section discusses our empirical results, and the final section rounds up with some conclusions and directions for future work.

The ADHOC Classification Method

ADHOC is a generic adaptive classification heuristic that maintains a bounded number of opponent classes $\mathcal{C} = \{c_1, \dots c_k\}$ for a society of agents $A = \{a_1, \dots a_n\}$ together with a (crisp) agent-class membership function $m : A \to \mathcal{C}$ that denotes which class any known agent a_j pertains to from the perspective of a modelling agent a_i. It is assumed that some OMM is used to derive a particular model of opponent behaviour c and that this model can be used to derive an optimal strategy for a_i when interacting with agents that pertain to this class. We further assume that interaction takes place between only two agents at a time in discrete encounters $e = \langle (s_0, t_0), \dots (s_l, t_l) \rangle$ of length l where s_i and t_i are the actions of a_i and a_j in each round, respectively. Each pair (s_i, t_i) is associated with a real-valued utility u_i for agent a_i[2].

The top-level ADHOC algorithm operates on the following inputs:

- an opponent a_j that a_i is currently interacting with,
- the behaviour of both agents in the current encounter e (we assume that ADHOC is called after the encounter is over) and
- an upper bound k on the maximal size of $\mathcal{C}$.

It maintains and modifies the values of

- the current set of opponent classes $\mathcal{C} = \{c_1, \dots c_{k-d}\}$ (initially $\mathcal{C} = \emptyset$) and
- the current membership function $m : A \to \mathcal{C} \cup \{\perp\}$ (initially undefined ($\perp$) for all agents).

Thus, assuming that an OMM is available for any $c = m(a_j)$ (obtained via the function $OM(c)$) which provides a_j with methods for optimal action determination, the agent can use that model to plan its next steps.

Similarity Measure and OPTALTCLASS procedure

The algorithm strongly relies on the definition a similarity measure $S(a, c)$ that reflects how accurate the predictions of c regarding the past behaviour of a are. In our prototype implementation, the value of S is computed as the ratio between the number of encounters with a correctly predicted by the class and the number of total encounters with a (where only *entirely* correctly predicted action sequences count as "correct" predictions).

[2]Any encounter can be interpreted as a fixed length iterated two-player normal-form game (Fudenberg & Tirole 1991), but the OMM we use here does *not* require that u_i be a fixed function that returns the same payoff for every enactment of a joint action (s_l, t_l).

This similarity measure is primarily used to decide whether to alter the classification of an agent, and also to determine the best alternative classification in that case, a task which is performed by the OPTALTCLASS procedure shown in Algorithm 1.

The OPTALTCLASS procedure proceeds as follows: if $\mathcal{C}$ is empty, a new class is created whose opponent model is consistent with e (function NEWCLASS). Else, a set of maximally similar classes $\mathcal{C}_{max}$ is computed, the similarity of which with the behaviour of a_j must be at least b (we explain below how this bound is used in the top-level heuristic).

If this set is empty, a new class is generated for a_j if the size of $\mathcal{C}$ does not exceed k. Else, OPTALTCLASS is called with $b = -\infty$, so that the side-condition of $S(a_j, c) \geq b$ can be dropped if necessary.

If $\mathcal{C}_{max}$ is not empty, i.e. there exist several classes with identical (maximal) similarity, we pick the best class according to the heuristic function QUALITY, which may use any additional information regarding the reliability or computational cost of classes.

In our implementation, this function is defined as follows:

$$\begin{aligned} \text{QUALITY}(c) &= \alpha \cdot \frac{\#\textit{CORRECT}(c)}{\#\textit{ALL}(c)} + \beta \cdot \frac{\#\textit{corr}(c)}{\#\textit{all}(c)} \\ &+ \gamma \cdot \frac{\#\textit{agents}(c)}{\#\textit{known\_agents}} \\ &+ (1 - \alpha - \beta - \gamma) \cdot \frac{1}{\text{COST(C)}} \end{aligned}$$

where

- $\#\textit{ALL}(c)$ is the total number of all predictions of class c in all past games,
- $\#\textit{CORRECT}(c)$ is the total number of correct predictions of class c in all past games,
- $\#\textit{all}(c)$ is the total number of all predictions of class c in the current encounter,
- $\#\textit{correct}(c)$ is the total number of correct predictions of class c in the current encounter,
- $\#\textit{agents}(c) = |\{a \in A | m(a) = c\}|$,
- $\#\textit{known\_agents}$ be the number of known agents,
- COST(C) is a measure for the size of the model $OM(c)$ and
- $\alpha + \beta + \gamma \leq 1$.

Thus, we consider those classes to be most reliable and efficient that are accurate in past and current predictions (usually, $\alpha < \beta$), that account for a great number of agents, and that are small in size.

Top-level heuristic

Given the OPTALTCLASS function that provides a mechanism to re-classify agents, we can now present the top-level ADHOC heuristic. In addition to the inputs and output data structures already described in the previous paragraphs, it uses the following internal parameters:

Algorithm 1 procedure OPTALTCLASS

```
inputs: Agent $a_j$, Encounter $e$, Set $\mathcal{C}$, Int k, Int $b$
outputs: Class $c$
begin
if $\mathcal{C} \neq \emptyset$ then
  {Compute the set of classes that are most similar to $a_j$, at least with similarity $b$ }
  $\mathcal{C}_{max} = \{c|S(a,c) = \max_{c' \in \mathcal{C}} S(a,c') \wedge S(a,c) \geq b\}$
  if $\mathcal{C}_{max} \neq \emptyset$ then
    {Return the "best" of the most similar classes}
    return $\arg\max_{c \in \mathcal{C}_{max}}$ QUALITY($c$)
  else
    {Create a new class, if $|\mathcal{C}|$ permits; else, the "high similarity" condition is dropped}
    if $|\mathcal{C}| < k$ then
      return NEWCLASS($e$)
    else
      return OPTALTCLASS($\mathcal{C}, k, -\infty$)
    end if
  end if
else
  return NEWCLASS($e$)
end if
end
```

- an *encounter comprehension* flag *ecf(c)* that is true, whenever the opponent model of some class c "understands" (i.e. would have correctly predicted) the current encounter;
- an "unchanged" counter $u(c)$ that counts the number of past encounters (across opponents) for which the model for c has remained stable;
- a *model stability threshold* τ that is used to determine very stable classes;
- *similarity thresholds* δ, ρ_1 and ρ_2 that similarities $S(a,c)$ are compared against to determine when an agent needs to be re-classified and which classes it might be assigned to.

During a given encounter with opponent a_j, it proceeds as presented in the pseudo-code description of Algorithm 2.

At the beginning, we set the current class c to the value of the membership function for the opponent a_j. Then, we update all classes' current similarity values with respect to a_j as described above, i.e. by dividing the number of past encounters with a_j that would have been correctly predicted by class c ($correct(a_j,c)$) by the total number of past encounters with a_j ($all(a_j)$).

If a_j has just been encountered for the first time, $m(a_j)$ is undefined ($c = \perp$) , and a_j is put into the best class that correctly predicts the data in the current encounter e. Since only one sample e is available for the new agent, setting $b = 1$ in OPTALTCLASS amounts to requiring that candidate classes correctly predict e. However, this condition is dropped inside OPTALTCLASS, if necessary. In that case, that class will be chosen for which QUALITY(c) is maximal.

So for the case of encountering a new agent, the algorithm ensures that the agent is either assigned to a (reasonably general and cheap) model that is consistent with current experience if one such model exists or that it is used to form a new category unless no more "agent types" can be stored.

Algorithm 2 ADHOC top-level heuristic

```
inputs: Agent $a_j$, Encounter $e$, Integer $k$
outputs: Set $\mathcal{C}$, Membership function $m$
begin
$c \leftarrow m(a_j)$
{The similarity values of all classes are updated depending on their prediction accuracy regarding $e$}
for all $c \in \mathcal{C}$ do
  $S(a_j,c) \leftarrow \frac{correct(a_j,c)}{all(a_j)}$
end for
if $c = \perp$ then
  {Unknown $a_j$ is put into the best sufficiently similar class that understands at least $e$, if any; else, a new class is created, if $k$ permits}
  $m(a_j) \leftarrow$ OPTALTCLASS($\mathcal{C}, k, a_j, 1$)
  if $m(a_j) \notin \mathcal{C}$ then
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{m(a_j)\}$
  end if
else
  {$c$ is incorrect wrt $a_j$ or very stable}
  if $S(a_j,c) \leq \delta \vee u(c) \geq \tau$ then
    {Re-Classify $a_j$ to a highly similar $c$, if any; else create a new class if $k$ permits}
    $m(a_j) \leftarrow$ OPTALTCLASS($\mathcal{C}, k, a_j, \rho_1$)
    if $m(a_j) \notin \mathcal{C}$ then
      $\mathcal{C} \leftarrow \mathcal{C} \cup \{m(a_j)\}$
    end if
  else
    {The agent is re-classified to the maximally similar (if also very stable) class}
    $c' \leftarrow$ OPTALTCLASS($\mathcal{C}, k, a_j, \rho_2$)
    if $c' \in \mathcal{C} \wedge u(c') > \tau$ then
      $m(a_j) \leftarrow c'$
    end if
  end if
  OM-LEARN($m(a_j), e$)
  {Model of $m(a_j)$ was modified because of $e$}
  if $ecf(m(a_j)) =$ true then
    {Reset similarities for all non-members of $c$}
    for all $a' \in A$ do
      if $m(a') \neq c$ then
        $S(a',c) \leftarrow 0$
      end if
    end for
  end if
  $\mathcal{C} \leftarrow \mathcal{C} - \{c'|\forall a.m(a) \neq c'\}$
end if
```

Next, consider the interesting case in which $m(a_j) \neq \perp$. In this case, we enter the re-classification routine to improve the classification of a_j is possible. To this end, we choose to assign a new class to a_j, if the similarity between agent a_j and its current class c falls below some threshold δ or if the model c has remained stable for a long time ($u(c) \geq \tau$) (which implies that it is valuable with respect to predictions about other agents). Also, we require that candidate classes for this re-classification be highly similar to a_j ($b = \rho_1$). As before, if no such classes exist, OPTALTCLASS will generate a new class for a_j, and if this is not possible, the "high similarity" condition is dropped – we simply *have* to classify a_j one way or the other.

In the counter-case (high similarity and unstable model), we still attempt to pick a new category for a_j. This time, though, we only consider classes that are very stable, very similar to a_j ($\rho_2 > \rho_1$), and we ignore classes output by OPTALTCLASS that are new (by checking "**if** $c' \in \mathcal{C} \ldots$"). The intuition behind this is to merge similar classes in the long run so as to obtain a minimal $\mathcal{C}$.

After re-classification, we update the class $m(a_j)$ by calling its learning algorithm OM-LEARN and using the current encounter e as a sample. The "problem case" occurs if e has caused changes to model c because of errors in the predicted behaviour of a_j (*ecf*$(m(a_j)) =$ true), because in this case, the similarity values of $m(a_j)$ to all agents are no more valid. Therefore, we choose to set the similarities of all non-members of c with that class to 0, following the intuition that since c has been modified, we cannot make any accurate statement about the similarity of other agents with it (remember that we do not store past encounters for each known agent and are hence unable to re-assess the values of S). Finally, we erase all empty classes from $\mathcal{C}$.

What this heuristic actually does is to create new classes for unknown agents or to put them into suitable classes if creating new ones is not admissible. After every encounter, the best candidate classes for the currently encountered agent are those that are able to best predict past encounters with it that have been stored in S. At the same time, good candidates have to be models that have been reliable in the past and low in computational cost. As far as action selection is concerned, a twofold strategy is followed: in the case of known agents, agent a_i simply uses $OM(m(a_j))$ when interacting with agent a_j, and the classification procedure is only called *after* an encounter e has been completed. If an unknown agent is encountered, however, the most suitable class is chosen for action selection in each *turn* using OPTALTCLASS. This reflects the intuition that the agent puts much more effort into classification in case it interacts with a new adversary, because it knows very little about that adversary.

Although all this may look fairly simple from a theoretical point of view, it turns out to be very effective in practice, even given a naive choice of the COST and S functions and a fairly *ad hoc* determination of the parameter values δ, ρ_1, ρ_2 and τ.

In the following section we turn to the scenario we have chosen for an empirical validation of the heuristic and to the details of combining ADHOC with an OMM for a given application domain.

ADHOC **in Multiagent IPD Simulations**

The scenario we have chosen to test the adequacy of our approach is fairly common and well-studied in the field of MAS research: the Prisoners' Dilemma (Luce & Raiffa 1957), a normal-form game for two players with action options C (cooperate) and D (defect) for both players and a payoff distribution as shown in Table 1. More precisely, our version of the multiagent Iterated Prisoners' Dilemma is based on a toroidal grid within which n agents move randomly and play with peers they happen to be in the same caret with (if more than two agents are co-located, all of them play with each other in a random order). Such a game consists of l iterations, and a total of N steps (meaning discrete moves on the grid) are executed in one simulation (where the number of iterated games depends on how often agents meet).

a_i \ a_j	C	D
C	(3,3)	(0,5)
D	(5,0)	(1,1)

Table 1: Prisoners' Dilemma payoff matrix. Matrix entries (u_i, u_j) contain the payoff values for agents a_i and a_j for a given combination of row/column action choices, respectively.

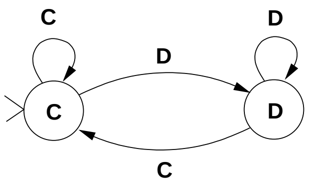


Figure 1: A DFA representing the TIT FOR TAT strategy in the PD game. Edge labels represent "own" action choices and state labels the other's reactions to these actions.

This simulation scenario, albeit simple, is complex enough to illustrate the necessity of strategy adaptation for every agent, and it also bears a potential for cooperation that has to be exploited if self-interested agents intend to collaborate effectively. Also, its simplicity enables us to conduct simulations with large numbers of agents, so that the emergence of "social complexity reduction" can be verified (if $n \gg k$ does not hold, one model can easily be maintained for every peer).

As far as finding an adequate opponent modelling method for the task of learning to play an IPD optimally against a single opponent is concerned, a combination of the DFA learning algorithm proposed by Carmel and Markovitch (Carmel & Markovitch 1996b; 1996a) and standard Q-learning (Watkins & Dayan 1992) is chosen, the details of which we can only sketch here for lack of space. During any encounter e, a_i attempts to learn the automaton that represents the strategy of a_j, where states are labelled with the actions of a_j and state transitions depend on a_i's own actions. Figure 1 depicts one such automaton for the famous TIT FOR TAT (Axelrod 1984) strategy.

Apart from the data stemming from e, the DFA stores a set of past encounters with that same class to provide for sufficient learning data.

In parallel with the evolution of the DFA, a table of Q-values is learned whose state space is the current set of DFA states. Optimal action selection then depends on the current state as tracked by the DFA of $c(a_j)$ during an encounter with a known agent a_j (with additional Boltzmann exploration to guide the search).

In the case of encountering a *new* agent for which no DFA and Q-table are available, a modified version of the OPTALTCLASS function is used, where $S(a, c)$ is defined in terms of correctly vs. incorrectly predicted *turns* (i.e. individual actions) rather than entire encounters. This modification is necessary, since during the first iterated game with a stranger the modelling agent does not possess data regarding entire encounters.

Also, the first action in an encounter with an unknown agent is chosen randomly, and this is also the case if OPTALTCLASS outputs a new class that is only consistent with a small initial portion of this encounter.

Of course, there exists a wide range of alternatives to this opponent modelling and strategy selection algorithm, e.g. to model opponents in a probabilistic fashion. However, the chosen OMM is capable of learning an adequate representation of the opponent's strategy in principle, and it meets certain criteria that should be warranted by any method that is to be used in combination with the ADHOC heuristic:

1. *The OMM should be capable of learning a model that adequately represents the opponent's strategy and which can be used to derive an optimal policy for the modelling agent.*

   In our scenario, this can be ensured as long as the representational power of deterministic automata is not exceeded by opponents. Further, the use of reinforcement learning provides safe convergence characteristics so that, if the learned DFA is correct, the modelling agent will converge to an optimal policy.

2. *Defining a similarity measure between any agent and any existing opponent model should be feasible.*

   Here, we have used a very simple method to derive the values for $S(a, c)$ that is based on weighing correct against false behaviour predictions.

3. *Opponent models should allow for the definition of a "cost" function.*

   In our case, the cost is simply defined as the number of DFA states, so as to prefer simpler automata for re-classification.

Experimental Results

We have conducted two series of experiments with the present implementation: first, simulations in which an ADHOC agent plays against a number of opponents with fixed strategies: "ALL C" (always cooperate), "ALL D" (always defect), "TIT FOR TAT" (cooperate in the first round; then play whatever the opponent played in the previous round) and "TIT FOR TWO TATS" (cooperate initially; then, cooperate iff opponent has cooperated in the two most recent moves). These strategies can be represented by very simple automata, and hence these simulations served as a starting point to verify whether the ADHOC agent was capable of performing the task in principle, i.e. to generate four classes and to converge to optimal strategies against all of them.

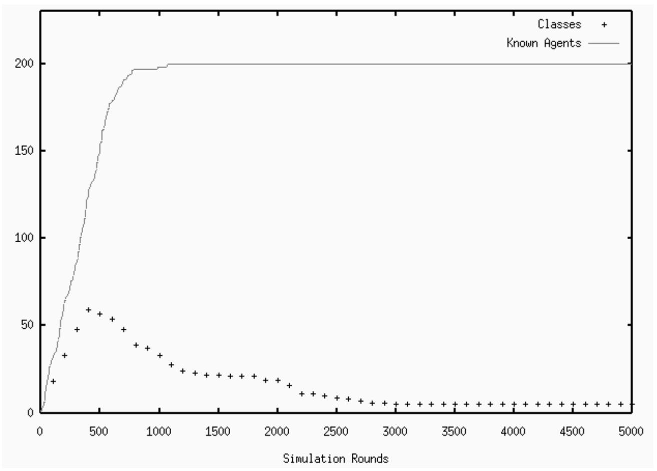


Figure 2: Number of agent classes an ADHOC agent creates over time (bold line) in contrast to the total number of known (fixed-strategy) opponents (which is increased by 40 in rounds 150, 300 and 450). The number of identified classes converges to the actual (four) strategies.

To obtain an adequate and fair performance measure for the classification heuristic, we compared the performance of the ADHOC agent to that of an agent who learns one model for every opponent it encounters and to that of an agent who learns a single model for *all* opponents.

The results of these simulations are shown in Figures 2 and 3[3]. They prove that the agent is indeed capable of identifying the existing classes, and that convergence to a set of opponent class is robust against entry of new agents into the system.

This is an important scalability result, because it means that in the long run, the agent exhibits satisficing behaviour even in very large-scale MAS, as long as the set of possible *strategies* is relatively small.

Next, let us turn to performance results. Here, interestingly, the ADHOC agent not only does better than an agent who maintains a single model for all opponents (which is easy to understand), but also significantly *outperforms* an allegedly "unboundedly rational" agent that is capable of constructing a new opponent model for each adversary – even though it is steadily increasing, that agent's performance remains below that of the ADHOC agent even after 40000 encounters. The reason for this is that the ADHOC obtains much more learning data for every class model it maintains by "forcing" more than one agent into that model, thus being able to learn a better strategy against every class within a shorter period of time.

This nicely illustrates another aspect of "social complexity reduction": the ability to adapt to adversaries quickly by virtue of creating "stereotypes".

[3]Each graph depicts the average of 100 simulations on a 10×10-grid. Parameter settings where: $\delta = 0.3$, $\tau = 15$, $\rho_1 = 0.6$, $\rho_2 = 0.9$, $k = 80$ and $l = 10$. 6 samples where stored for each class in order to learn automata.

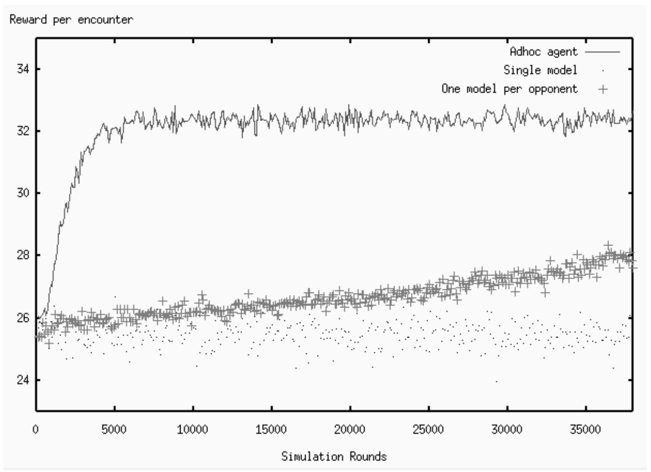
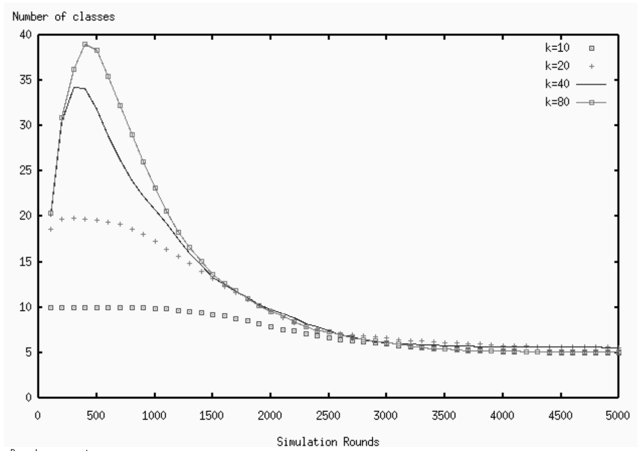


Figure 3: Comparison of cumulative rewards between ADHOC agent, an agent that maintains one model for each opponent and an agent that has only a single model for all opponents in the same setting as above.

An issue that deserves analysis is, of course, the appropriate choice of the upper bound k for the number of possible opponent classes. Figure 4 shows a comparison between ADHOC agents that use values 10, 20, 40 and 80 for k, respectively, in terms of both number of opponent classes maintained and average reward per encounter. Quite surprisingly, even though there seems to be not much difference between the time-spans that are necessary to converge to the optimal number of opponent classes, there seem to be huge differences with respect to payoff performance. More specifically, although a choice of $k = 40$ instead of $k = 80$ seems to have little or no impact on performance, values of 10 and 20 are certainly too small. How can we explain this result? On the one hand, it is certainly the case that, the more models are maintained, the more exploration will be carried out per model in order to learn an optimal strategy against it. On the other hand, the fewer models we are allowed to construct, the more "erroneous" will these models be in predicting the behaviour of adversaries that pertain to them (until we gather enough training data), since we are not allowed to make many distinctions.

Although it is certainly true that we have to trade off these two aspects against each other (both extremely high and extremely low values for k seem to be inappropriate), our results here (that are in favour of large values for k) *contrast* the previous observation that "creating stereotypes" increases efficiency. Allowing some diversity *during* learning seems to be crucial to achieve effective learning and interaction.

In the second series of experiments, we conduct simulations with societies that consist entirely of ADHOC agents. With the algorithms presented in the previous sections, agents seem to exhibit random behaviour throughout these simulations. This is, in fact, quite understandable considering that they have fairly random initial action selection distributions (when Q tables are not filled yet and automata unsettled) and that, hence, no agent can strategically adapt to the strategies of others (since they do not have a strategy, either).



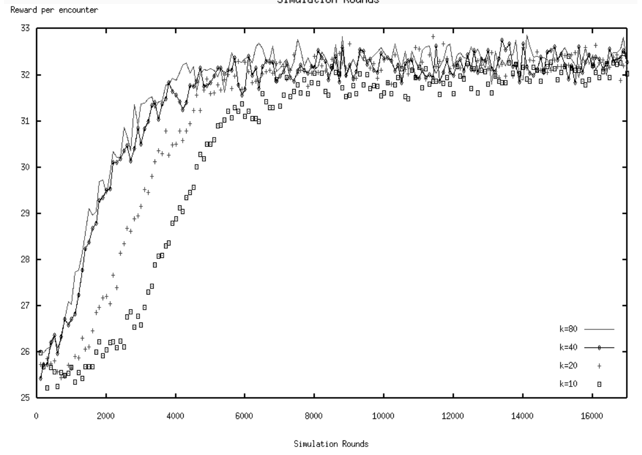


Figure 4: Comparison between ADHOC agents using different k values. The upper plot shows the number of opponent classes the agents maintain, while the plot below shows the average reward per encounter. The number of opponent classes remains stable after circa 5000 rounds.

With a slight modification to the OMM, however, we obtain interesting results: we simply add the following rule to the decision-making procedure:

> *If the automaton of a class c is constantly modified during r consecutive games, we play some fixed strategy X for r' games; then we return to the strategy suggested by $OM(c)$.*

Applying this rule implies that if an agent identifies a complete lack in strategy on the side of its peers, it takes the initiative to "come up" with one (being "creative"), maintain it for a while, and the return to adaptation. The results of these simulations, where X =TIT FOR TAT are shown in Figure 5, and illustrate that ADHOC here seems to be capable of exploiting, at least partially, the cooperation potential that exists in the IPD (the sum of agents' payoffs clearly exceeds that of a random mixed strategy combina-

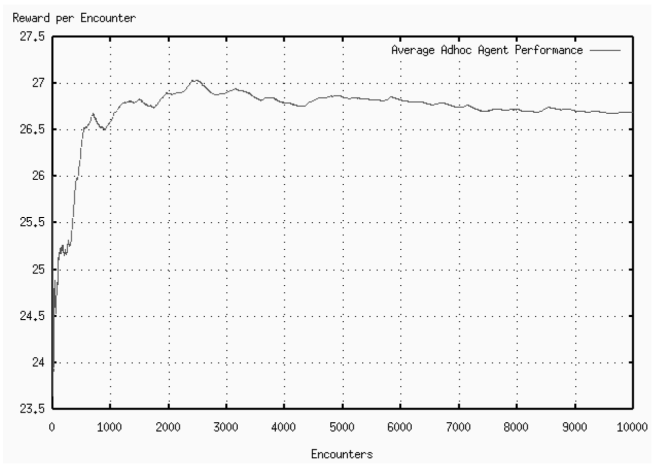


Figure 5: Average payoff per encounter (10 PD iterations) among 10 ADHOC agents. It lies clearly above 25 (=(50+0)/2 for (C,D) combinations) but below 30 (=(30+30)/2 for (C,C) combinations).

tion ($1/4 \cdot 10 + 1/4 \cdot 0 + 1/4 \cdot 3 + 1/4 \cdot 5 = 22,5$), yet without being able to guarantee the establishment of stable cooperation, either.

Although X was hand-crafted here and although this last result will surely require further investigation, we believe that adding additional reasoning capabilities to the OMM is possible in order to identify and maintain such "adaptation-viable" strategies and this is an issue that will have to be looked at in the future.

Still, our experiments prove that ADHOC is at least capable of "trying out" new strategies without jeopardising long-term performance.

Conclusion

This paper summarises initial work on a new opponent classification heuristic called ADHOC which is based on the intuition of "categorising" adversaries in multiagent interactions so as to achieve *social complexity reduction* in open MAS that consist of self-interested, potentially malevolent agents. To our knowledge, the work presented here constitutes the first attempt to tackle the problem of opponent modelling in the face of occasional encounters in large-scale multiagent systems.

First empirical results prove the adequacy of our approach for domains where agents with regular strategies interact in discrete two-player encounters. We showed how boundedly rational ADHOC agents outperform "unboundedly rational ones", which they achieve by combining learning samples from all class members during opponent (class) modelling. At the same time, the experiments suggest that best performance results can be achieved if we allow for a great number of classes in the early phases of learning, so that an initial lack in the *quality* of the models can be alleviated by using a great *variety* of sub-optimal models. More generally, we might infer from this observation that using stereotypes is only highly effective if those stereotypes are good – otherwise, it is better to rely on modelling individuals considered different from each other.

As far as interaction *among* ADHOC agents is concerned, we are currently working on a more elaborate and more general heuristic than the one presented above (i.e. occasionally playing fixed, hand-crafted strategies). The central problem we are faced with here is the fact that ADHOC agents cannot be modelled as DFAs themselves. First experiments with heuristics that are based on the idea of sticking to own past behaviours in case the opponent exhibits random behaviour (in order to promote the establishment of stable interaction patterns) appear to produce promising results, and deserve further research.

Many issues remain to be examined in the future: first of all, some limitations of the algorithm need to be done away with, most importantly: the restriction to "post-encounter classification". Ideally, we would like to classify constantly throughout the encounter in order to fine-tune classification choices and to have the *option* of whether to modify a model or to re-classify the agent. Further, restricting ourselves to two-agent interactions is clearly unrealistic with respect to real-world applications. Also, domains more complex than that of IPD should be examined, and generic rules for the choice of ADHOC parameters should be derived. Surely, complex domains will also require more elaborate similarity and model cost measures.

Secondly, the very important issue of *partner selection* should be looked at, since dealing with various types of peer agents will normally imply that interactions with them will differ in usefulness (and these differences, again, may depend on the current goals of the agent). Therefore, it is only natural to consider agents that have the ability to *choose* which agent they interact with, at least under certain conditions. This will not only have effects on the performance of ADHOC agents, but also on the learning and classification process as such.

Finally, as we are looking at *social* phenomena, the potential of using communication to achieve better coordination has to be explored – since messages are actions with little cost and no immediate utility, their availability is expected to increase the adaptability and strategic rationality of intelligent, opponent classifying ADHOC agents vastly.

References

Axelrod, R. 1984. *The evolution of cooperation*. New York, NY: Basic Books.

Balch, T. 1997. Learning roles: Behavioral diversity in robot teams. In *Collected Papers from the AAAI-97 Workshop on Multiagent Learning*. AAAI. 7–12.

Bui, H.; Kieronska, D.; and Venkatesh, S. 1996. Learning other agents' preferences in multiagent negotiation. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 114–119. Menlo Park, CA: AAAI Press.

Carmel, D., and Markovitch, S. 1996a. Learning and using opponent models in adversary search. Technical Report 9609, Technion.

Carmel, D., and Markovitch, S. 1996b. Learning models of intelligent agents. In *Thirteenth National Conference on Artificial Intelligence*, 62–67. Menlo Park, CA: AAAI Press/MIT Press.

Freund, Y.; Kearns, M.; Mansour, Y.; Ron, D.; Rubinfeld, R.; and Shapire, R. E. 1995. Efficient Algorithms for Learning to Play Repeated Games Against Computationally Bounded Adversaries. In *36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, 332–343. Los Alamitos, CA: IEEE Computer Society Press.

Fudenberg, D., and Tirole, J. 1991. *Game Theory*. Cambridge, MA: The MIT Press.

Luce, R. D., and Raiffa, H. 1957. *Games and Decisions*. New York, NY: John Wiley & Sons.

Rovatsos, M., and Lind, J. 2000. Hierarchical common-sense interaction learning. In Durfee, E. H., ed., *Proceedings of the Fifth International Conference on Multi-Agent Systems (ICMAS-00)*. Boston, MA: IEEE Press.

Russell, S., and Wefald, E. 1991. *Do the right thing: Studies in limited rationality*. Cambridge, Mass.: The MIT Press.

Stone, P., ed. 2000. *Layered learning in multiagent systems. A winning approach to robotic soccer*. Cambridge, MA: The MIT Press.

Vidal, J. M., and Durfee, E. H. 1997. Agents learning about agents: A framework and analysis. In *Collected papers from AAAI-97 workshop on Multiagent Learning*. AAAI. 71–76.

Watkins, C., and Dayan, P. 1992. Q-learning. *Machine Learning* 8:279–292.