Towards Social Complexity Reduction in Multiagent Learning: The ADHOC method

Michael Rovatsos and Marco Wolf
{rovatsos,wolf}@informatik.tu-muenchen.de

Computer Science Department
Technical University of Munich
Motivation

- Open multiagent systems (MAS)
  - necessity of modelling peer agents to achieve coordination
- Large-scale MAS imply only occasional encounters with acquainted peers
  - acquiring and maintaining information about individual agents hard and/or inefficient
- Is employing models of whole classes/types of other agents a solution?
- Objective: application of the principle of social complexity reduction to artificial agent societies
Overview

1. Introduction
2. The AdHOC Heuristic
3. Application to Multiagent IPD Games
4. Experimental Results
5. Conclusions & Outlook
Introduction
Introduction

• In human societies reducing the number of models of others is a common phenomenon, e.g.
  - roles in organisations,
  - stereotypes,
  - legal regulations,
  - ...

• Learning opponent models is a prominent issue in multiagent learning

• Applying classification techniques to such opponent models has not received much attention

• Our aim: to combine opponent modelling techniques with classification
Introduction (contd.)

- Advantages: Using a limited number of models
  - reduces computational cost,
  - is adequate for modelling *bounded rationality*,
  - speeds up learning of the models by increasing learning data.

- Assumptions:
  - no prior knowledge about others’ goals or strategies,
  - other’s strategies need not be fixed over time,
  - no benevolence assumptions, no common goals,
  - active, on-line learning during interaction.
The AdHOC Heuristic
The **ADHOC** heuristic

- **ADaptive Heuristic for Opponent Classification**
- Evolves up to \( k \) opponent classes \( C \) with a (crisp) membership function \( m : A \rightarrow C \) for an arbitrary set of opponents \( A \).
- Assumes an underlying Opponent Modelling Method (OMM) that returns an opponent model \( OM(c) \) for each class and which
  - is capable of adequately describing the opponent’s behaviour,
  - allows for computing the similarity \( S(a, c) \) between peer \( a \) and opponent class \( c \),
  - makes it possible to determine an optimal behaviour towards class \( c \).
The AdHoc heuristic (II)

The heuristic

- processes data observed during encounter $e = \langle (s_0, t_0), \ldots (s_l, t_l) \rangle$ with utility $u_i(s_l, t_l)$ for the modelling agent $a_i$,
- determines the optimal class for $a_j$ while constantly updating similarity values $S(a_j, c)$ for all classes,
- adapts the model of a class through use of the OMM for agent $a_j$ in case of weak similarity,
- handles similarity values in case of model adaptation.

Central sub-procedure: OPTALTCLASS
The ADHOC heuristic (III)

procedure OPTALTCLASS\( (a_j, e, \mathcal{C}, k, b) \)

if \( \mathcal{C} \neq \emptyset \) then

\( \mathcal{C}_{max} \leftarrow \{ c \mid S(a_j, c) \text{ maximal } \land S(a_j, c) > b \} \)

if \( \mathcal{C}_{max} \neq \emptyset \) then

return \( \text{arg max}_{c \in \mathcal{C}_{max}} \text{QUALITY}(c) \)

else

if \( |\mathcal{C}| < k \) then

return NEWCLASS\( (e) \)

else

return OPTALTCLASS\( (a_j, e, \mathcal{C}, k, -\infty) \)

end if

end if

else

return NEWCLASS\( (e) \)

end if
The AdHoc Heuristic (IV)

- Heuristic “quality” function in our implementation:

\[
\text{QUALITY}(c) = \alpha \cdot \frac{\#\text{CORRECT}(c)}{\#\text{ALL}(c)} + \beta \cdot \frac{\#\text{correct}_c}{\#\text{all}(c)} + \gamma \cdot \frac{\#\text{agents}(c)}{\#\text{known\_agents}} + (1 - \alpha - \beta - \gamma) \cdot \frac{1}{\text{COST}(C)}
\]

- \text{OPTALTCLASS} is used throughout the top-level heuristic to find the most appropriate class for a peer \(a_j\).
The ADHOC Heuristic (V)

\quad \text{procedure ADHOC}(a_j, e, k) \\
\quad \quad \forall c \in C. S(a_j, c) \leftarrow \text{correct}(a_j, c) \div \text{all}(a_j) \\
\quad \quad \text{if } m(a_j) = \bot \text{ then } m(a_j) \leftarrow \text{OPTALTCLASS}(a_j, e, C, k, 1) \text{ else} \\
\quad \quad \quad \text{if } m(a_j) \text{ doesn’t predict } e \text{ correctly then} \\
\quad \quad \quad \quad \text{if } S(a_j, m(a_j)) \leq \delta \text{ } \& \text{ } m(a_j) \text{ is very stable then} \\
\quad \quad \quad \quad \quad m(a_j) \leftarrow \text{OPTALTCLASS}(a_j, e, C, k, \rho_1) \\
\quad \quad \quad \text{end if} \\
\quad \quad \quad c' \leftarrow \text{OPTALTCLASS}(a_j, e, C, k, \rho_2) \\
\quad \quad \text{if } c' \in C \wedge c' \text{ is very stable then } m(a_j) \leftarrow c' \\
\quad \quad \text{OM-LEARN}(m(a_j), e) \\
\quad \quad \text{if } m(a_j) \text{ has been modified then} \\
\quad \quad \quad \forall m(a') \neq m(a_j). S(a', m(a_j)) \leftarrow 0 \\
\quad \quad \text{end if} \\
\quad \text{end if} \\
\text{end if}
The ADHOC Heuristic (VI)

Control flow during encounters:

1. Action selection:
   (a) If $a_j$ is encountered for the first time, \textsc{OptAltClass} is called \textit{after each turn} to determine the most suitable class.
   (b) Else, $OM(m(a_j))$ is used throughout the encounter.

2. After the encounter is over, the classification procedure is called.

3. Empty classes are erased from $C$. 
Application to Multiagent IPD Games
Application scenario (I)

- Well-understood application example: Iterated Prisoner’s Dilemma games.
- Payoff matrix:

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>(3,3)</td>
<td>(0,5)</td>
</tr>
<tr>
<td>D</td>
<td>(5,0)</td>
<td>(1,1)</td>
</tr>
</tbody>
</table>

- Simulations consist of fixed-length IPD games between randomly matched agents on a toroidal grid (with random agent movement).
Application scenario (II)


- Model opponents as deterministic finite automata (DFA): transitions depend on own actions, states are labeled with other’s actions
- Lookup table for Q-values uses DFA states as MDP states
- Boltzmann exploration:

$$P(z) = \frac{e^{Q(s,z)} / T}{\sum_{z'} e^{Q(s,z')} / T}$$
Application scenario (III)

Properties of the opponent modelling method:

- Guaranteed to converge to a DFA consistent with the peer automaton
- Can be easily combined with RL methods
- Similarity function is easy to define by checking correct prediction of an interaction sequence
- Models cannot be improved incrementally
- Limited expressiveness, esp. it does not cater for non-deterministic behaviour
Experimental Results
Experimental results (I)

Simulation settings:

- ADHOC agents play against fixed-strategy opponents with the following (and random DFA) strategies:

  - TFT
  - TF2T
  - AllC
  - AllD

- Typically, we increase population from 80 to 200 agents over time (tested populations up to 1000)
- Store 6 encounter samples per class for DFA learning
AdHOC Performance

![Graph showing AdHOC Performance](image-url)
Different $k$ values – Rewards

Interactions (IPD Games)

Reward per 100 Games

$k = 80$  
$k = 60$  
$k = 40$  
$k = 30$  
$k = 20$  
$k = 10$
Different $\kappa$ values – Classes
**AdHOC vs. AdHOC**

- AdHOC agents do well against fixed-strategy opponents
- They learn faster than “unboundedly rational” agents
- They can manage with low values of $k$

*but*

- Playing against other AdHOC agents causes random behaviour
- Reason: learning agents cannot be represented by DFA
**AdHOC vs. AdHOC**

- Extend the heuristic to detect random opponent behaviour
- In that case, switch to a fixed DFA according to which the agent will play (for a limited period of time)
- Strategies for selecting this DFA:
  - use a hard-coded strategy (e.g. TIT FOR TAT),
  - choosing a random automaton from \( C \),
  - choosing the DFA with maximum expected payoff/quality.
- Results rather unsatisfying
AdHOC vs. AdHOC – Rewards

Interactions (IPD Games)

Reward per 100 Games

- Tit For Tat
- Random Class
- Maximum Quality
- Highest Payoff
AdHOC vs. AdHOC – Classes

![Graph showing encounters vs. number of opponent classes for different strategies: Tit For Tat, Random Class, Maximum Quality, and Highest Payoff.]
Conclusions
Conclusions

• Achieved social complexity reduction by applying a simple classification heuristic
• Leads to considerable speed-up in the convergence of models
• Seamless integration of learning, classification and strategic reasoning
• Paves the way for boundedly rational yet effective adaptation in large-scale open multiagent systems
• Problem: adaptive, classifying agents cannot be represented with the opponent modelling method used here
Outlook

- Model adaptation *during* encounters
- Richer interaction scenarios (esp. partner selection and context-dependent strategy choice)
- Explore possibilities mixed or fuzzy class membership functions
- Use of explicit communication for higher-level coordination
- Alternative opponent modelling methods to cope with ADHOC vs. ADHOC (incremental, probabilistic)
Thank you for your attention!