Towards Any-Team Coaching in Adversarial Domains

Patrick Riley
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213-3891
pfr@cs.cmu.edu

Manuela Veloso
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213-3891
mmv@cs.cmu.edu

Gal Kaminka
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213-3891
galk@cs.cmu.edu

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence; I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems—*Games*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Experimentation, Standardization

1. THE COACHING PROBLEM

As multi-agent systems continue to grow in importance, the types of relationships between agents continue to be studied. One important relationship that humans often exhibit is a coach. For example, the lead programmer in a software development team provides structure, direction, and a problem decomposition to the other programmers and a professor provides guidance and advice to her graduate students in search of their Ph.Ds.

In order to explain the problem a coach faces, one must first define what the *role* of a coach is. A coach is a member of a team in the sense of having a common goal. However, in common usage (such as used in sports), a distinction is usually made between the coach and the team of players. We preserve this common usage here, and discuss a single coach working with multiple teams of agents. Unlike other agents in the team, the coach's only action is to communicate to the agents on the team, which we will call the receivers. The coach's goal is to improve the performance of the team through this communication.

The communications from the coach should suggest changes to the receivers' behavior. The expressiveness and flexibility of communication languages can vary greatly. Advice can be very specific, such as "In this state, take this action" or very general, such as "Your goal should now be this." Also, for more general advice, the coach may want the agents to be independent and not follow advice in all situations.

Given the explanation of the coach role, the coaching problem can be stated quite simply: "How can an agent in a coach role improve the performance of the team?" The coach agent may be a separate agent whose only role is coach, or a team member may fulfill the coach role in addition to others. For example, the lead programmer in a software development team may write code, as well as communicating with and coordinating the other programmers.

AAMAS'02, July 15-19, 2002, Bologna, Italy.

There are many reasons one might want to consider a coach role in a team of autonomous agents. A coach role provides a method of oversight for the agents and can aid in the creation of agents with adjustable autonomy [5]. By separating the coach from the details of an environment (by forcing only communication from the coach role), the problem should encourage reuse of algorithms across environments. Also, the coach problem provides a decomposition of the problem of performing well in an environment. Good decompositions are critical for complex environments.

We now consider in more detail why a coach agent may be able to provide benefit to a team of receptive agents. We identify some possible features of the environment or execution features which suggest that coaching may be useful.

Computation power A coach could provide a logical breakdown for distributing computation.

Central information gathering point If the coach is given a more complete view, some of this information can be shared with the agents. Alternatively, the agents can communicate local state information with the coach, who transmits condensed advice to the other agents.

Adversarial information point In environments with non-cooperative or adversarial agents, the coach provides a single point of entry for information about such agents.

Coordination by authority The coach can provide efficient coordination by avoiding costly and potentially slow negotiation. How to do so while maintaining sufficient flexibility for the agents is an interesting problem.

Focused learning The coach can provide a middle ground between centralized and distributed learning by providing guidance to the learning agents.

In the design of a coach agent, one of the first questions is what part the of performance of the team the coach wants to change. Below, we identify several areas in which a coach can provide advice.

Local adjustment Here the coach identifies small changes which could affect the performance of the receivers, in the same spirit as hill-climbing methods.

Opponent analysis Opponent modelling can provide benefit to a team of agents, and relevant information about those models can be communicated to the agents.

Adaptation The coach may enable the receivers to respond more quickly to changes in the environment either by having more information or processing power, or simply by providing coordination.

In order to further explain the nature of the coaching problem, we identify a few general techniques below.

Information summarization If the coach has more information or processing power, useful summaries of the events in the environment could be provided.

Experiment and learn If there is sufficient time, the coach can experiment with advice and gather training data on it's effect.

Planning The many variants of planning have the potential to provide both coordination and lookahead to the receivers.

Imitation If a well-performing group of agents are available to be studied, the coach can provide advice to try to imitate the performance of the groups.

2. COACHING IN ROBOTIC SOCCER

We have implemented a coach using the Soccer Server System [2] as used in RoboCup [1]. Our coach uses planning and imitation to analyze the opponent and provide fast adaptation. Because of the creation of a standard coach language, coaches are able to communicate with several teams of simulated soccer playing agents. This was the basis for a small coach competition at RoboCup2001.

The language CLang [4] was adopted as a standard language for a coach competition at RoboCup2001. Basically CLang provides advice in the form of condition-action rules. The conditions are over the state of the world and the actions are meta-actions which may take several lower level actions to accomplish. Four teams competed providing a unique opportunity to see the effects of a coach designed by one group on the team of another.

We participated in the coach competition, which consisted a single game in each test case. This section reports on our thorough empirical evaluation of the teams involved in the competition. An analysis of the techniques our coach uses can be found in [3]. Each experimental condition was run for 30 games and the average score difference (as our score minus their score) is reported. All significance values reported are for a two tailed t-test.

We will use initials to denote the teams. The four teams that understand CLang are: WrightEagle (WE), HelliRespina (HR), DirtyDozen (DD), and ChaMeleons (CM). We prepend "C-" to indicate the coach from that team. Our experiments pair up the various coachable teams and coaches against a fixed opponent, Gemini (GEM). Team descriptions for these teams are available in [1]. We also wrote a coach which sends random advice by combining primitives used by other coaches.

The results can be seen in Table 1. The score differences in the upper table are relative to the score difference of each team without a coach, shown in the lower table. The intervals are 95% confidence intervals. An 'X' in a location indicates that we were unable to those experiments because of technical difficulties.

First, for the WE row, none of the differences between the entries are significant ($p > .17$ for all pairs). We hypothesize that WE is in fact effectively ignoring what the coach says since even the random coach has no significant effect on the players.

	Coaches			
Team	Random	C-HR	C-DD	C-CM
WE	1.0 [0.1,1.9]	0.2 [-1.0,1.4]	0.7 [-0.2,1.7]	0.0 [-1.0,1.1]
HR	X	0.0 [-0.5,0.5]	-3.2 [-4.0,-2.4]	0.2 [-0.2,0.8]
DD	X	X	-1.4[1] [-2,7,-0.1]	8.4 [7.6,9.3]
CM	-8.3 [-9.3,-7.3]	2.1 [1.4,2.8]	1.3 [0.7,1.9]	4.4 [4.0,4.9]

WE	HR	DD	CM
9.1 [8.1,10.2]	1.6 [1.1,2.1]	-17.2[-18.1,-16.3]	-6.5 [-7.2,-5.9]

Table 1: Score differences (positive is winning) for four teams and coaches.

For the HR row, it is clear that the team is listening to the advice, because C-DD has a highly significant ($p < .000001$) negative effect on the team. However, both the C-HR (the coach designed for that team) and our coach C-CM have no significant effect on the score ($p > .44$). Skipping down to the CM row, all the coaches have a significant positive effect on the team CM ($p < .005$), with our coach C-CM (the coach designed with CM) having the greatest effect.

For the DD row, the notable effect is the large goal change (+8.4) for the team DD using our coach C-CM. Even though the team and the coach were not designed together, our coach can help their team. For the rest of the C-CM column, our coach helps CM ($p < .000001$), and causes no significant effect on the other two teams ($p > .44$).

Clearly, coaches in this domain can have a both a positive and negative impact on the performance of a team. Notably, our coach never hurts the performance of a team and can improve performance.

3. CONCLUSION

We have presented a general description of the coaching problem. We believe the coaching problem can provide a good way to decompose the goal of achieving good performance for agents in many domains, especially multi-agent and adversarial ones. Further, we have presented empirical results from a simulated robotic soccer domain. Using a standard coaching language, teams and coaches not designed together are able to function together. This research is a first step in understanding advice-based relationships between automated agents. Many interesting questions are raised which we will continue to pursue.

4. REFERENCES

[1] A. B. S. Coradeschi and S. Tadokoro, editors. *RoboCup-2001: Robot Soccer World Cup V.* Springer Verlag, Berlin, 2002.

[2] I. Noda, H. Matsubara, K. Hiraki, and I. Frank. Soccer server: A tool for research on multiagent systems. *Applied Artificial Intelligence*, 12:233–250, 1998.

[3] P. Riley, M. Veloso, and G. Kaminka. An empirical study of coaching. In *Distributed Autonomous Robotic Systems 6*. Springer-Verlag, 2002. (to appear).

[4] RoboCup Federation, `http://sserver.sourceforge.net/`. *Soccer Server Manual*, 2001.

[5] P. Scerri, D. Pynadath, and M. Tambe. Adjustable autonomy in real-world multi-agent environments. In *Agents-2001*, 2001.

[1] It should be noted than one agent on the DD team crashed each time this experiment was run, so it may not be meaningful to compare these results to the others shown here