Coordination Advice: A Preliminary Investigation of Human Advice to Multiagent Teams

Nathan Schurr
University of Southern California
Powell Hall of Engineering 514
Los Angeles, CA 90089-0781
schurr@usc.edu

Paul Scerri
Carnegie Mellon University
Pittsburgh, PA
pscerri@cs.cmu.edu

Milind Tambe
University of Southern California
Powell Hall of Engineering 208
Los Angeles, CA 90089-0781
tambe@usc.edu

Abstract

This paper introduces a new area of advice that is specific to advising a multiagent team: Coordination Advice. Coordination Advice differs from traditional advice because it pertains to coordinated tasks and interactions between agents.

Given a large multiagent team interacting in a dynamic domain, optimal coordination is a difficult challenge. Human advisors can improve such coordination via advice.

This paper is a preliminary look at the evolution of Coordination Advice from a human through three different domains: (i) disaster rescue simulation, (ii) a self-maintaining robotics sensors, and (iii) personal assistants in a office environment. We study how the useful advice a person can give changes as the domains change and the number of agents and roles increase.

Introduction

Teams of intelligent members are emerging as an exciting paradigm for achieving complex tasks in dynamic, distributed environments. These teams may be comprised of agents, robots or even people, and must coordinate despite uncertainty, spatial distribution and limited communication bandwidth (Pynadath & Tambe 2003; Scerri *et al.* 2003). In order to overcome these challenges, the team of robots is guided by a Team Oriented Program (TOP) (Pynadath *et al.* 1999; Tidhar 1993) that describes their plan and overarching goals.

The challenge of performing effective coordination (Pynadath & Tambe 2002) requires that the team use heuristic algorithms of reasonable complexity, which may turn out to be suboptimal. Unfortunately, due to its inability to observe the entire state of the whole world, an agent team will often not be able to detect when its approximations are not working effectively and its behavior is unnecessarily poor. Even if a team becomes aware of its poor performance, it might not know how to remedy its position. One approach to improving a team's functioning is to have a human watch the execution of the team and provide advice when required to improve the performance. The idea of advice works best when the human has a reasonable overview of team performance.

The goal of our research is to develop general methods for building advice-taking teams, and this paper presents our framework and some initial experiments. Various techniques have been applied to the general problem of allowing human reasoning to improve agent behavior. This idea of human advice has been explored in the field of reinforcement learning (RATLE) (Maclin & Shavlik 1996) and single agent planning (Myers 1996). However, previous techniques have been mostly applied to the case of advising a single agent. There has been some work on advising teams, such as the work at CMU where a software agent coach advises a simulated soccer team (Riley & Veloso 2001). Our focus is on a human advisor to a team of agents. Two limitations with this previous work become apparent when considering the case of advising a team of agents. First, these prior methods have worked with advice that is only given offline. This does not suit the dynamic and unpredictable environments that we are dealing with because the context of the situation, which is critical to the advice, is not known in advance. Myers' Advisable Planning Systems require advice prior to the formation of the plans (Myers 1996). Even in RATLE, execution of the reinforcement learning is stopped, advice is factored in and then execution can resume. Second, previous approaches do not address the issue of advising a distributed team of agents, but instead focus on advising an individual agent. However, in a team setting, advice pertaining to coordination among multiple agents becomes necessary.

Thus, in this paper we propose that a new type of advice arises when a human is to give advice to an entire team of multiagents. We call this new category Coordination Advice. We describe three examples of domains where we have applied Coordination Advice and discuss why the approaches differed. First, in a disaster rescue simulation, we discuss the notion of advice on allocating individual roles. Second, where a group self-maintaining robotics sensors is advised on how to prioritize the roles that are to be done. Third, we explore the domain of a team of personal assistants in a office environment and begin to start giving advice on which role allocation algorithms to choose. Each of these represent a different level of Coordination Advice that was necessary for the scale and domain.

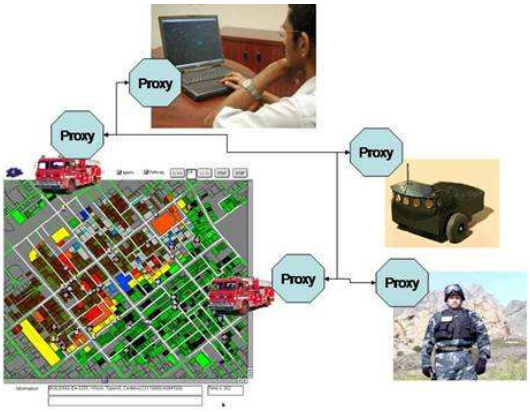

Figure 1: Disaster Rescue Architecture.

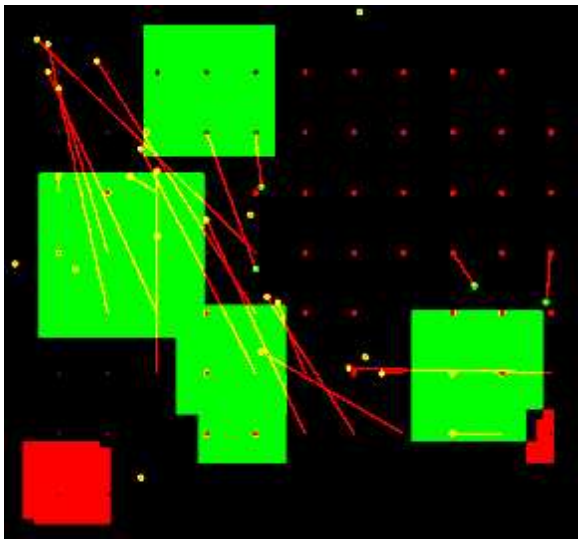

Figure 2: Screenshot of Simulator.

Domains

Here we will define the domains which we will be focusing on. First, a quick overview of the disaster rescue domain. Then we will discuss more in depth a self-maintaining sensor network. Finally we will introduce an office assistant domain.

In all three domains, we assume a large-scale team of agents that are following a Team Oriented Program (TOP) (Pynadath *et al.* 1999; Tidhar 1993). We assume complete communication between agents and the advisor in order to concentrate on the issue of how to interpret advice. However there are limits to the amount of bandwidth and knowledge that the agents have. All agents are assumed to be collaborative and coordinate based on heuristics that are aimed at benefitting the whole team, but need not. Although advice can conceivably stem from a vast array of sources, we focus on how to interpret advice that comes from humans. For our preliminary study, we will assume that the human advisor has complete, accurate information about the state of the team. The human is not necessarily an expert of the domain and needs no knowledge of the heuristic model or TOP that the team is following. Instead, the advisor has a general knowledge of what the team's abstract goals are and what is used as a measure of the team's performance.

Disaster Rescue

A human advisor takes the position of a fire chief and interacts with a team of fire engine agents that are fighting fires in a simulated disaster (see Figure 1). The human fire chief advises the team by taking on the task of assigning roles that the team cannot allocate itself. For a more in-depth description of the disaster rescue scenario and our implementation, please refer to our previous paper (Scerri *et al.* 2003).

Robotic Sensors

The simulator we use replicates a situation in which a team of self-maintaining robotics sensors must perform over an extended period of time. Throughout this time a set of robots make sure that the sensor network stays intact while the batteries of the sensors are continually being depleted and some sensors even break down and fail. In order to keep sensors from running completely out of battery, this team of robots can each take a single sensor at a time to a charging station to replenish the battery. Then the sensor must be placed back into the physical network. The robots that move the sensor around can be thought of as shepherds moving around the sensors (sheep). For example, when a sheep is out of battery, a shepherd will move the sheep to the recharging station and back again to its original location (home). At any one time, a particular shepherd robot will be assigned to the role of recharging a single sheep sensor. After the shepherd's sheep has been recharged and returned to its home, that shepherd may then switch to the role of recharging any available sheep. The shepherds adhere to a TOP that they all possess. This TOP explains to the group how it will complete the recharging process on a particular sheep as well as the method that each shepherd will implement when choosing which sheep to begin recharging. Figure 2 shows a screenshot of this simulator, in which the shepherds (bright dots) are trying to pick up the sheep that are arranged in a grid (dark dots). Once a shepherd has been assigned to a pick up a sheep, there is a line drawn from the shepherd's current location to the sheep's home. The brighter colored rectangles are regions of advice and will be explained further in section .

Personal Assistants

Last we deal with an office environment where there is a group of people in the same research group. Each person has an agent that represents them and facilitates that person participating in team plans. Each agent has a copy of the same TOP (see Figure 3). We deal with the general plan of giving a presentation together. A presentation consists of multiple roles that should be allocated to different members

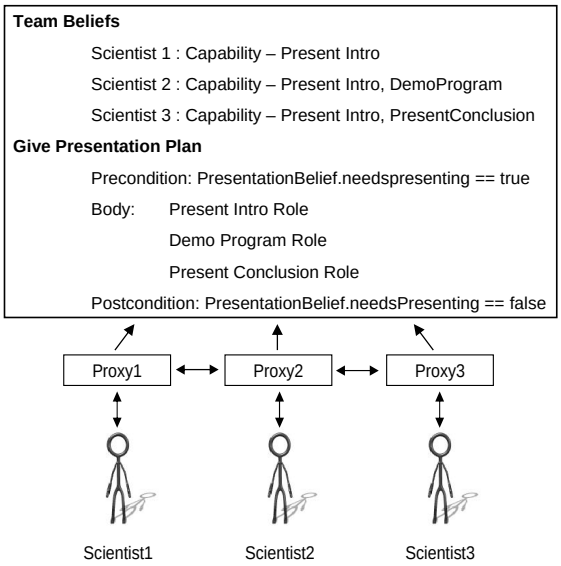

Figure 3: Office assistant TOP and architecture.

of the group. The team plan is instantiated once the belief exists that there is a presentation that needs to be done. Only one agent considers taking on a role at a time in order to eliminate redundancy of plan roles. At the time of consideration, the agent can either ask the person it represents if that role should be taken or the agent can decide autonomously whether or not the role should be accepted. It determines this by estimating a capability level of the person, based on the person's ability to do the task and how many roles that person currently has. If that capability level is higher than a threshold that is set for that particular role, the agent accepts the role and notifies the person. Otherwise, the role is rejected and passed on to another agent in the hopes of it being allocated to someone more capable.

Multiagent Aspects

Coordination of multiple agents is the primary concept that comes into play when dealing with large teams and complex tasks. Coordination Advice has the potential to arise even in the shepherd and sheep scenario mentioned earlier. Though not currently implemented in this way, imagine that the task of picking up a sheep is a coordinated role that requires two shepherds. These two shepherds are needed in order for each one to hold an opposite end and pick up the sheep. Here lies an example of where Coordination Advice can be a suggestion of how many shepherds should pick up a sheep at the same time. In this case, unless two shepherds pick up a particular sheep in a coordinated manner, the efforts of the shepherds are futile. Furthermore, sending too many shepherds at once would be a waste of both resources and time. This complex problem of having a decentralized team of agents work together in order to act at coinciding times is an aspect that is unique to advice-taking teams.

Critics might argue that there is nothing different that comes up when considering a large team of agents, and the same issues would come up when giving advice to a single agent. Actually, giving advice to these large teams brings up new issues that do not arise when dealing with giving advice to a single agent primarily because of some key traits that exist when considering a team with a large number of agents.

Taking a more detailed view, this coordination of large-scale teams has three main implications. One implication is that the advice about coordination is different from other kinds of advice. This is because Coordination Advice inherently involves a concept that is dependant on more than one agent. Consequently, the team must agree on the validity and implications of the advice. Thus, team members may need to negotiate in order to operationalize the advice in real-time, as no single agent could operationalize it alone. Otherwise the advice could have a great potential for negatively impacting the team. The second implication is that if we were to advise a team on coordinated action by advising individuals, it would be more tedious. An advisor can either advise the team together or give advice to each individual on what it should do. In large-scale teams, addressing the whole team is easier, while addressing individual members may require too much micro-management. Providing advice at the level of the team when doing coordinated action is somewhat easier for the human than providing direct advice to individual agents. The third implication of team-level advice is that while it is useful, it means that the advice is not tailored to each individual. Yet the advice must still must have effects on an individual agent level. Thus, given team advice, each individual has to interpret that advice in its own context and act in the best interest of the team. For instance, if the team is advised to make a certain role a top priority and everyone in the team rushes to do the same task, severe problems will occur. At the same time, some members of the team, while not performing the task, should know that it has been made a priority because of their current task's effect on the prioritized task.

This suggests that the advice at the team level may need to be high-level advice and it may need to be more abstract than at the single agent level. Within this new Coordination Advice type, lies the ability to affect not only what a team does, but also how a team works together. Furthermore, it allows the human advisor to give compact advice, without unnecessarily dealing with the intricacies of the team's actions.

Approach

Coordination Advice is advice that can be given on the team level that applies to team tasks (roles). Much in the same way that a fire chief possessed a good ability to allocate roles in the disaster rescue simulations, humans have often have a good grasp of the larger picture. As the scale of the team and dynamics of the environment increase, it becomes impossible for the human advisor to allocate each role. This is why in our previous experiments we shifted into a region-based task advice which allowed the advisor to offer advice on a general area of the map. Even this approach has its limits once we scale up even further and the advisor cannot

Domain	*Disaster Rescue*	*Robotic Sensors*	*Personal Assistants*
Advice Instigator	Agent	Human	Human
Advice Content	Role Allocation Roles	Region-Based Role Priorities	Role Allocation Method
Scale (Agents,Roles)	(10,20)	(30,100)	(100,1000)
1 Piece of Advice	Single Role	Region of Roles	All Tasks (Potentially)

Table 1: Coordination Advice Across Multiple Domains.

offer advice that is helpful for an entire region of discernable tasks. We argue that as these complex multiagent teams surface, the solution to allowing effective human interaction with the multiagent team in order to increase performance or efficiency will be with respect to the way the team coordinates rather than on specific decisions that individuals make.

As seen in Table 1, we have applied Coordination Advice in three different domains using three different approaches in order to allow the human advisor to interact with the different multiagent teams. In the disaster rescue scenario, an agent instigates the advice by offering roles that the team of fire engines cannot allocate to the human advisor (fire chief). The fire chief is then free to offer advice by assigning roles to team members as he sees fit. This approach resulted in an increase in performance of the fire engines. However as seen in the Results section below (see Table 2 and 3), as the numbers of agents increased, the amount of help that the human advisor could offer was less. We believe this is due to the fact that a piece of advice only affected a single agent to fulfill a single role. Therefore, as the domain scaled up in the number of agents and roles, the need arises for more abstract levels of Coordination Advice.

We considered this when approaching the next domain where we allowed a human advisor to help a team of self-maintaining sensors scan an area while not running out of batteries. Here we allowed the human to instigate the advice by drawing rectangular regions over the map and affecting the priority of roles in that region. This allowed for a single piece of advice to affect multiple roles. This lessened the burden on the advisor by allowing advice to affect general regions. However, the advisor was sometimes ineffective due to being bogged down by having to make arbitrary local decisions and merely identifying problematic areas.

This influenced our approach to the final domain: personal assistants. Our prior robotic sensor team experiments showed us that humans are often not better than the agents themselves at making the decisions about details. Yet, they are good at choosing the basic approach for a team to take. So we allowed the human advisor to give advice to the team on which role allocation method to use. With the potential for a lot more roles in this domain, this category of Coordination Advice allows the human advisor to affect the whole team's handling of roles in a more abstract way rather than being burdened by the details of each role.

Results

In this section, we will cover the results of three sets of experiments. The first experiments involve an disaster rescue domain where a human advisor (fire chief) is allowed give advice by allocating roles (Scerri *et al.* 2003). Then we will discuss experiments that were performed in the robotic sensors (shepherd and sheep) domain where a human advisor is allowed to give advice on the priority of a general area of tasks. Finally we will show results from an office assistant scenario in which a human is allowed to advise on how to implement the role allocation.

Disaster Rescue

In previous experiments, we focus on an disaster rescue scenario in which a human fire chief interacts with a team of fire brigades in order put out fires that are popping up all over the city. The fire chief interface consists of two frames. One frame shows a map of the city, displaying labelled markers for all of the fires that have been found, the positions of each fire brigade, and the location of the role each fire brigade is assigned to. The fire chief does not have direct access to the state of the simulator. Instead, the fire chief is updated by the messages received through the fire chief's proxy. Therefore, the fire chief may be viewing a delayed picture of the simulation's progress. The other frame displays a list of all of the role allocation tasks that have been allocated to the fire chief. By clicking on a task, the relevant capability information about each fire brigade is shown. The right-side window lists the fire brigades' distances to the fire, their water levels, and the roles they are currently performing. The fire chief can then view this data and find an appropriate agent to fulfill the role.

We conducted tests with three different fire chiefs. Each completed several practice runs with the simulation prior to experiments in order to minimize any learning effects. Each scenario was run for 100 time steps, with each step taking 30 seconds. The total data presented here represents 20 hours of run-time with a human in the loop.

Table 2 shows the team's domain-level performance across each experimental configuration. The scoring function measures how much of the city was destroyed by fire, with higher scores representing worse performance. The table shows the mean scores achieved, with the standard deviations in parentheses. Examining our two dimensions of interest, we can first compare the two rows to examine the effect of increasing the complexity of the coordination problem. In this case, increasing the number of fire brigades improves performance, as one might expect when adding resources while keeping the number of initial tasks fixed.

However, we can dig a little deeper and examine the effect of increasing complexity on the fire chief's performance. In the simpler configuration, asking the fire chief earlier (i.e.,

# Brigades	*maxAsked*= 0%	*maxAsked*= 100%	*maxAsked*= ∞
3	58(3.56)	73(16.97)	74(0.71)
10	52(19.09)	42(14.00)	73(4.24)

Table 2: Domain-level performance scores.

# Brigs.	*max Asked*	Domain Roles	Fire Chief Roles	Tasks Performed	% Tasks Performed
3	0%	116 (7.12)	401 (51.81)	27 (6.55)	23.29 (6.51)
	100%	146 (33.94)	407 (54.45)	24 (6.36)	16.02 (0.63)
10	0%	103 (38.18)	864 (79.90)	67 (2.83)	14.49 (2.13)
	100%	98 (42.40)	563 (182.95)	41 (8.38)	48.06 (19.32)

Table 3: Role and fire-chief task metrics.

maxAsked= 0) improves performance, as the team gets a head start on exploiting the person's capabilities. On the other hand, in the more complex configuration, asking the fire chief earlier has the opposite effect. To better understand the effect of varying the point at which we assign roles to people, Table 3 presents some of the other statistics we gathered from these runs (mean values, with standard deviations in parentheses). With 3 brigades, if we count the mean number of roles taken on by the fire chief, we see that it stays roughly the same (401 vs. 407) across the two *maxAsked* settings. In this case, asking the fire chief sooner, allows the team to exploit the person's capabilities earlier, without much increase in his/her workload. On the other hand, with 10 brigades, the fire chief's mean role count increases from 563 to 716, so although the proxies ask the fire chief sooner, they are imposing a significant increase in the person's workload. Judging by the decreased average score in the bottom row of Table 2, the increased workload more than offsets the earlier exploitation of the person's capabilities. Thus, our experiments provide some evidence that increasing domain-level scale has significant consequences on the appropriate style of interaction with human team members.

Robotic Sensors

In order to further leverage human advice, we set out to find out how the human advisor could help a team of robotic sensors that need to be shepherded back to their recharging station. For these initial experiments, we simulated the sheep and shepherd scenario (mentioned earlier) with a 30 shepherd team responsible for 80 sheep placed in a 100 by 100 square room. Each battery starts out with a value of 1.0 and at every time step the each sheep's battery value is decreased by a value of .00125. Once a sheep's battery level reaches zero, the sheep is considered to be dead and unrecoverable (unable to charge again). Thus a sheep could die in 800 steps. A shepherd is allowed to move in one of the 4 primary directions a distance of 1 unit in the room each time step. Experiments were run over a 2500 time step period. The primary metric that we used to measure the team's overall performance was how many of the 80 sheep ended up dying by the end of the scenario's duration.

The human has the ability to give advice to the team by increasing or decreasing the priority of the sheep in a selected region. A region of unlimited size can be selected by an advisor using the mouse, pressing to start expanding region, and releasing to stop. We performed three tests each with the human advice weight (h) set to both 1.0 and 0.5. We ran the same configuration 30 times without any advice (human advice weight (h) set to 0).

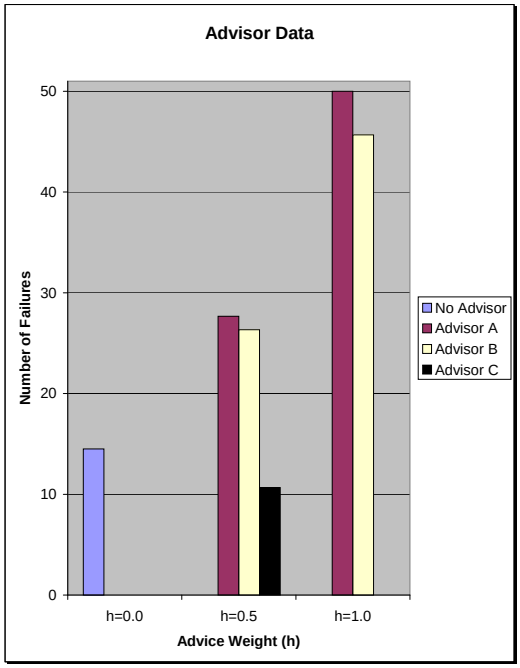

Figure 4: Advisor Performance (number of failures).

# of Agents	Advice	Messages	Allocation Success
3	No	1120	No
3	Yes	9	Yes
7	No	2570	No
7	Yes	21	Yes

Table 4: Comparison after 5 minutes of trying to find an allocation.

Given this experimental setup, without any advice, an average of around 15 sheep died by the end of the experiment (see Figure 4). However, as seen in the graph, the human advisors on average didn't help the number of sheep deaths decrease. In fact, after being advised the team of shepherds often ended up doing worse. Yet, this is not to say that augmenting the team's optimized heuristic in the manner that we provided was unable to improve performance at all. Advisor C actually managed to find a strategy during his experiments that allowed him to average of about 10 sheep dying.

Personal Assistants

We created a scenario in which a team of agents were collaborating in the office environment. These office assistant agents would help to allocate roles to the person that they represent. In our scenario we had these agents allocate tasks necessary for combining efforts in order to make presentations. If the capability of the person is above the threshold necessary to take on this role, the role is accepted, otherwise it is rejected and passed on. In our experiments, we inundate the team with several presentations that result in all team members becoming busy and therefore having a low capability for performing a role. Then we introduce a new presentation role into the team. This role has a threshold that is higher than can be found now in this team. The human can see this situation will arise, and under tightly constrained systems like this one, can offer advice to alleviate unnecessary role passing. The human advisor was allowed to give advice to disregard the thresholds policy in the standard role allocation algorithm and take on any task for which the person has a capability. The human advisor was allowed to give this advise to a team of 3 and a team of 7 agents. The advice resulted in a change in the way that the role allocation algorithm behaved. As can be seen in Table 4, this advice caused an allocation to occur, though a sub-threshold allocation, whereas before no allocation would be made. In addition, far fewer messages are passed back and forth in the 5 minutes we allowed for the teams to run for. Even with advice, each agent receives 3 messages in both the 3 and 7 agent experiments, which result in 9 and 21 total messages respectively (see Table 4). This is due to agents communicating between each other in order for the team to agree on the beliefs that a presentation must be made, there is a plan to present for that presentation, and the role has been assigned (yet below threshold due to the advice).

Summary

This is a preliminary investigation into Coordination Advice, which can be used when dealing with multiagent systems. The goal of Coordination Advice is to give a human advisor the means of advising a multiagent team that is collaborating in a dynamic, complex environment. All of this must be done without the advisor being overwhelmed. We have applied Coordination Advice to three different domains, each where a human advisor has been able to interact with these multiagent teams. These domains are disaster rescue, sensor robotics and personal assistants. In these domains, we implemented varying levels of advice that cover a wide range of advice abstractions and present these results.

References

Maclin, R., and Shavlik, J. W. 1996. Creating advice-taking reinforcement learners. *Machine Learning* 22(1-3):251–281.

Myers, K. 1996. Advisable planning systems. In *Advanced Planning Technology*.

Pynadath, D. V., and Tambe, M. 2002. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research* 16:389–423.

Pynadath, D. V., and Tambe, M. 2003. Automated teamwork among heterogeneous software agents and humans. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)* 7:71–100.

Pynadath, D. V.; Tambe, M.; Chauvat, N.; and Cavedon, L. 1999. Toward team-oriented programming. In *Proceedings of Agent Theories, Architectures, and Languages Workshop*, 233–247.

Riley, P., and Veloso, M. 2001. Coaching a simulated soccer team by opponent model recognition. In *Proceedings of the Fifth International Conference on Autonomous Agents (Agents-2001)*. (extended abstract).

Scerri, P.; Pynadath, D. V.; Johnson, L.; P., R.; Schurr, N.; Si, M.; and Tambe, M. 2003. A prototype infrastructure for distributed robot-agent-person teams. In *The Second International Joint Conference on Autonomous Agents and Multiagent Systems*.

Tidhar, G. 1993. Team oriented programming: Preliminary report. In *Technical Report 41, Australian Artificial Intelligence Institute, Melbourne, Australia*.