Cooperation and Coordination Between Fuzzy Reinforcement Learning Agents in Continuous State Partially Observable Markov Decision Processes

Hamid R. Berenji, David Vengerov
Intelligent Inference Systems Corp.
Computational Sciences Division, MS: 269-2
NASA Ames Research Center
Mountain View, CA 94035
berenji@ptolemy.arc.nasa.gov
vengerov@stanford.edu

Abstract—

Successful operations of future multi-agent intelligent systems require efficient cooperation schemes between agents sharing learning experiences. We consider a pseudo-realistic world in which one or more opportunities appear and disappear in random locations. Agents use fuzzy reinforcement learning to learn which opportunities are most worthy of pursuing based on their promised rewards, expected lifetimes, path lengths and expected path costs. We show that this world is partially observable because the history of an agent influences the distribution of its future states. We consider a cooperation mechanism in which agents share experience by using and updating one joint behavior policy. We also implement a coordination mechanism for allocating opportunities to different agents in the same world.

Our results demonstrate that K cooperative agents each learning *in a separate world* over N time steps outperform K independent agents each learning in a separate world over K*N time steps, with this result becoming more pronounced as the degree of partial observability in the environment increases. We also show that cooperation between agents learning *in the same world* decreases performance with respect to independent agents. Since cooperation reduces diversity between agents, we conclude that diversity is a key parameter in the trade off between maximizing utility from cooperation when diversity is low and maximizing utility from competitive coordination when diversity is high.

I. INTRODUCTION

In recent years there has been a considerable amount of interest in multi-agent systems [e.g., Luck, 1997; Sycara, 1998]. One approach to modeling multi-agent learning is to augment the state of each agent with the information about other agents [Littman, 1994; Mataric, 1997; Stone and Veloso, 1999]. However, this approach is difficult to implement in noisy environments where agents cannot reliably discern states and actions of other agents. A more decentralized approach is to give each agent the capability of independent learning in the environment while allowing agents to share learning experience when they find it beneficial.

Several cooperative learning models of this type have been proposed. Kelly and Keating, [1998] considered situations where several robotic agents were learning to avoid stationary and moving obstacles. A common set of fuzzy automata was used by cooperating agents to represent their possible states. Learning consisted of agents taking turns in updating probabilities with which actions were selected in each automaton. In another related work, Tan [1993] simulated several hunters learning to capture a prey in a tile world using reinforcement learning (RL). He studied two RL-based cooperation mechanisms: agents updating a common policy and agents averaging periodically their individual policies. He found that both performed equally well on this task, outperforming uncooperative agents that were learning for the same number of time steps. Whitehead [1991] obtained a theoretical upper bound on performance improvement due to cooperation between multiple agents in Markov Decision Processes (MDP's). He showed that K agents learning over N time steps can perform at best as one agent learning over N*K time steps. This result was confirmed experimentally by Tan.

II. Overview

This paper extends the above results to Partially Observable Markov Decision Processes (POMDP's) with continuous state spaces. We consider a learning problem with a fixed mild non-Markovian character in state transitions and adjustable non-Markovian character in the step cost function. A stochastic and partially observable tile world is used as a testing domain in our experiments. This world basically consists of multiple opportunities of different rewards appearing in random locations and remaining there for a random period of time described by a known probability distribution. Each step in the world carries a variable cost depending on the roughness of the terrain. In this world, agents learn which opportunities are most worthy of pursuing based on their promised rewards, expected lifetimes, path lengths and expected path costs. A detailed description of this world is given in the next section.

In our first set of experiments, we analyze effects of several agents learning in the same world. In order to supervise the competition of agents for the existing opportunities, we introduce a coordination mechanism that is parameterized by a "selfishness" parameter S. This parameter controls the degree to which agents choose opportunities to pursue in order to maximize their own utility vs. choosing them to maximize the team performance.

The value of $S = 0.5$ implies a behavior for all agents where they weigh equally the desires of other agents. A value of $S = 1$ implies a behavior where each agent puts a weight of 1 on its own desires. We show that depending on the model parameters, there is an optimal value of S, $0 < S < 1$, that maximizes the total combined reward for all agents. We also give an explanation why this value is not 0.5.

In our second set of experiments, we implement one of the cooperation mechanisms considered by Tan [1993] that allows agents to share the learning experience. This is one of the simplest cooperation mechanisms in which several agents are using and updating the same set of Q-values. The other mechanism considered by Tan [1993] consisted of agents maintaining individual state values but averaging them out periodically. Tan found that both of these algorithms performed equally well.

In order to isolate effects of cooperation from those of coordination, we first consider having multiple worlds with the same statistical properties, each inhabited by a single agent. Our results show that cooperation between agents results in a significant performance improvement with respect to independent agents learning for the same number of time steps. This implies that in real world applications, several robotic agents can achieve much better results if they learn cooperatively for the allocated period of time instead of learning independently.

Unlike Tan [1993], we are able to take our cooperation results further and show that average performance of K cooperative agents each learning for N time steps in our domain is significantly higher than average performance of K independent agents each learning for K*N number of time steps, contrary to Whitehead's theoretical results for MDP's. We show that this performance improvement increases with an increasing level of partial observability present in the environment. These cooperation results suggest that cooperative performance is more robust than individual performance in the presence of difficulties in the learning problem such as nonstationarity and partial observability.

In our third set of experiments, we analyze effects of coordination on cooperative learning. We obtain an unexpected result that several agents learning cooperatively in the same world perform much worse than independent agents. At the same time, if we create two parallel worlds with the same statistical properties and allow agent i of world 1 cooperate with agent i of world 2, then cooperative agents again perform much better than independent agents. Since cooperation between agents in the same world reduces diversity between agents, we conclude that diversity is a key parameter that trades off maximizing utility from cooperation when diversity is low vs. maximizing utility from competitive coordination when diversity is high.

III. Domain Description

The world we have constructed is a variation of a tile world. A location in this world is given by a lattice point - a point with integer coordinates. At each time step, an agent chooses to move either straight or diagonally from its current location to one of the 8 adjacent locations.

Each step in the world has a certain cost, which is calculated using the potential field method: the cost of moving to any location is equal to the potential at the destination. The potential function is generated by randomly choosing locations of deformations of varying strength that generate a potential field. The potential at any location due to a certain deformation is given by

$$P = \frac{h}{(d + 0.5)^2}$$

where h is the height of the deformation and d is the distance to it. The potential at the location of that deformation is equal to h. Potentials due to all deformations add up to give the final potential at any location. An agent then multiplies this sum by a step cost scaling (SCS) parameter to obtain the cost of moving to the considered location. A deformation is relocated at each

time step with a small probability, and its value changes randomly during the relocation. Hence, agents are travelling through a constantly changing potential surface and do not specialize their policies to a particular pattern of deformations.

Opportunities that promise some rewards appear randomly in different locations. An opportunity can expire at time t with probability $1 - \exp^{\frac{-1}{M}}$ where M is its mean lifetime. If an agent moves to the location of an unexpired opportunity, it receives the reward promised by that opportunity minus the total path cost since the beginning of the episode. An opportunity disappears once it has been reached by an agent and a new opportunity appears in a randomly chosen location. After that, a new episode begins. If the opportunity towards which the agent took its last step expires, the agent obtains a negative reward equal to its path cost traveled so far and a new episode begins. The objective of an agent is to maximize the total reward minus the total cost received during the simulation.

Our tile world was designed to reflect the complex trade-offs that humans encounter in many decision situations. The notion of "opportunity" used in our domain description is equivalent to the notion of "alternative" in decision analysis. One of the main difficulties in applying decision analysis to complex situations is that of putting possible outcomes in the order of preference. In the world described above, the agent has to balance the reward of an opportunity against its mean lifetime to come up with some measure of reward the agent expects to receive. Then, the agent has to balance distance to this opportunity with roughness of the path towards it to come up with some measure of cost the agent expects to incur. Finally, the agent has to balance expected reward with expected cost to come up with the total desirability of the opportunity.

In the next section we show that the problem of learning the best opportunity to pursue in the described world is very difficult, as it creates an extremely large state space for the agent. However, we are able to find a state-space reduction that still allows agents to learn very good policies. Also, as will become apparent in the next section, the learning algorithm used by the agents does not rely on the presence of discrete tiles in the world and will work just as well in a continuous world. We used discrete tiles only for the ease of computer implementation.

IV. LEARNING ALGORITHM

When the state space in a reinforcement learning problem is large, learning separate Q-values for every state-action pair is very inefficient. In the early stages of learning it is very unlikely that the same state will be visited again, and hence the agent will always have to act based on ini-

tial Q-values. The standard approach for dealing with this problem is to generalize the Q-values across states by using a function approximation architecture $Q(i, a, r)$ for approximating $Q(i, a)$, where r is the set of all learned parameters arranged in a single vector. The basic parameter updating rule used by such a function approximation architecture is:

$$r_t \leftarrow r_t + \alpha \nabla_{r_t} Q(i, a, r_t) \delta_t, \qquad (1)$$

where α is the learning rate and δ is the Bellman error used in the corresponding learning rule for the look-up table case:

$$Q(i_t, a) \leftarrow Q(i_t, a) + \alpha \delta_t. \qquad (2)$$

For example, in the look-up table version of Monte Carlo learning,

$$\delta_t = \sum_{\tau=t}^{T} \gamma^{\tau-t} g(\tau) - Q(i_t, a), \qquad (3)$$

where $g(t)$ is the cost incurred at time t, γ is the discounting factor and the summation extends until the end of the episode. In the look-up table version of $TD(0)$ learning,

$$\delta_t = g(t) + \gamma max_a Q(i_{t+1}, a) - Q(i_t, a). \qquad (4)$$

In this paper we will use fuzzy state aggregation as the approximation architecture. That is,

$$Q(i, a) = \sum_{k=0}^{K} q(k, a) \mu_k(i_t), \qquad (5)$$

where $q(k, a)$ is the Q-value of taking the action a in the k-th fuzzy state s_k and $\mu_k(i_t)$ is the degree of membership of state i_t to s_k. If the action space is continuous, then equation (5) still applies after changing $\mu_k(i_t)$ to $\mu_k(i_t, a)$. Also note that equation (5) is linear in the learned parameters $q(k, a)$; hence, fuzzy state aggregation is just a linear approximation architecture.

With $Q(i, a)$ given by equation (5), equation (1) becomes a matrix equation with each component given by:

$$q(k, a) \leftarrow q(k, a) + \alpha \mu_k(i_t) \delta_t. \qquad (6)$$

Equation (6) has a natural interpretation in the realm of fuzzy state aggregations: the value of a fuzzy state-action pair (s_k, a) gets updated proportionally to its contribution to the Q-value of the state-action pair (i_t, a) in equation (5).

As will be described later, we found that the learning framework of average reward per time step is more appropriate for our domain than the discounted framework presented above. In this framework, equation (6) still holds, except that δ is given by [Sutton and Barto, 1998]:

$$\delta_t = \sum_{\tau=0}^{T} \gamma^{\tau-t} g(\tau) - Q(i_t, a) - \rho_t \qquad (7)$$

for Monte Carlo learning, and by

$$\delta_t = g(t) + \gamma max_a Q(i_{t+1}, a) - Q(i_t, a) - \rho_t \quad (8)$$

for $TD(0)$ learning. The quantity ρ represents the average reward per time step of the policy learned so far, to which the average reward from every state-action pair is compared. The quantity ρ is updated at every iteration according to

$$\rho_t \leftarrow \rho_t + \alpha\delta_t. \quad (9)$$

A. Coordination Algorithm

The purpose of the coordination algorithm used in this paper is to allocate existing opportunities to the agents. The coordination mechanism is initialized at every time step with all opportunities being open to all agents. Then the following process repeats.

REPEAT

Each agent considers its most preferred open opportunity and asks all other agents for their preferred opportunities and distances to them. If there is no conflict, the agent finishes happy and quits. If there is a conflict, the agent determines the potential winner (the closest agent or the one with the highest preference if there is a tie). If the agent is not the potential winner in a conflict, it indicates that it is unhappy and passes control to the next agent. If the agent is the potential winner , it performs the winner arbitration. If after arbitration it still holds the opportunity, the agent ends up happy and quits, if not - the agent indicates that it is unhappy and passes control to the next agent.

During the winner arbitration, the potential winner considers whether it should abandon the opportunity in favor of its second choice and give up the first choice opportunity to the next closest agent (or the one with the highest preference if a tie) that is open to the opportunity. The winner abandons the opportunity if

$$S\delta_1 - (1 - S)\delta_2 < 0,$$

where δ_1 is the agent's own δ (i.e., difference in preferences between the considered opportunity and the next most desirable that is still open) and δ_2 is the other agent's δ. If the winner abandons the opportunity, it closes itself to it. If the winner doesn't abandon the opportunity, it sends a signal to all other agents to close themselves to the opportunity.

UNTIL all agents are happy.

Note that even totally selfish arbitration with $S = 1$ still produces positive results of informing other agents about the uselessness of their attempts by closing off their first choice opportunities if there is a conflict.

B. Representation for Learning

The full state of an agent at time t can be described by: $i = (i_1, i_2, ..., i_P)$, where i_p is a group of five state variables describing the opportunity p. These variables are:

1. distance to the opportunity

2. reward of the opportunity

3. path roughness to the opportunity

4. mean lifetime of the opportunity

5. relative direction of the opportunity

Path roughness to the opportunity gives the agent an approximate measure of the average cost per step on the way to that opportunity. It is obtained by first constructing an ellipse with the major axis extending from the agent's current location to the location of the opportunity and passing some number of units beyond the opportunity. The path roughness is then calculated as the sum of the values of all deformations in that ellipse divided by the area of the ellipse.

The values of each state variable are described using a number of fuzzy labels, such as SMALL, MEDIUM, and LARGE. Each value can match one or more labels with a certain membership value. For example, if the opportunity rewards vary between 0 and 100 units, then the value of 30 can be SMALL to degree 0.7 and MEDIUM to degree 0.3, while the value of 10 can be SMALL to degree 1.0. A fuzzy state is represented by a collection of fuzzy labels, one for each state variable. For example, one of the states in the full learning problem can be: (distance to opportunity 1 is LARGE) AND (reward of opportunity 1 is LARGE) AND (mean lifetime of opportunity 1 is SMALL) AND ... . The degree to which an agent belongs to a certain fuzzy state is the minimum of the degrees to which all state variables belong to the labels describing this fuzzy state.

In the simplest case, each state variable is described by only two fuzzy labels: SMALL and LARGE. Then there are 2^{5P} possible fuzzy states for each agent. This seems to be an impractically large number of fuzzy states which defeats the purpose of significantly reducing the state space through state aggregation.

In order to avoid this exponential growth we restrict the Q-value of a state-action pair (i, p) only to be a function of (i_p, p). That is, when the agent evaluates action p of moving towards opportunity p, it can only observe the information about that opportunity. With this restriction an agent will not be able to learn that it is better to move towards an opportunity surrounded by other ones (in case the chosen opportunity disappears, there will be

other ones nearby) rather than moving towards an isolated opportunity. This state reduction also implies that the relative direction of each opportunity can be omitted from the state description if we specify that the agent will always go directly to the chosen opportunity. This will not change the nature of our experiments since the problem of choosing the direction of motion once an opportunity is chosen is very easy in comparison with the problem of choosing an opportunity to pursue.

The final learning problem becomes in essence the problem of evaluating 2^4 fuzzy states for each considered opportunity p. Then, in order to find the Q-value of the state-action pair (i, p), the agent will consider the four relevant state variables and evaluate the degree to which this quadruple belongs to each of the 16 fuzzy states. The agent will then use equation (5) for combining the Q-values of the fuzzy states to come up with the final Q-value.

The final learning problem faced by the agent is non-Markovian for two reasons. First of all, because of the state restriction, knowledge of the previous opportunities that the agent was pursuing affects the probability distribution of the future states. For example, if a better opportunity appears near the one the agent was pursuing, the agent will switch to pursuing it and all components of its state vector will change. However, if the new opportunity disappears very soon, then the agent will most likely switch back to pursuing its original opportunity. Therefore, remembering the characteristics of the original opportunity biases our beliefs about future states if the switching situation described above takes place. Second, the learning problem with using all $5P$ state variables is still non-Markovian because of the partial observability in the cost function that creates a history dependence. For example, knowledge of the trend in the costs incurred during the previous time steps affects the probability distribution of the future step costs because the pattern of surrounding deformations is likely to remain unchanged. Note that an increase in the step cost scaling parameter increases the importance of the partial observability of the state and consequently the degree to which the learning problem is non-Markovian.

V. Experimental Setup

We used a 20-by-20 tile world in all experiments. There were always 10 opportunities in the world, and whenever one of them would be reached by an agent or would expire, a new one would appear in a random location. The mean lifetime of each appearing opportunity was uniformly distributed between 5 and 20, and its value was uniformly distributed between 0 and 100. The values of appearing deformations were uniformly distributed between 0 and 100. Each deformation would get relocated with proba-
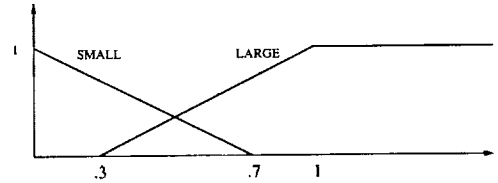

Figure 1: Fuzzy labels used by the agents

bility 0.01 and its value would randomly change in the process. There were 20 deformations in the world.

For faster simulations, we used only two fuzzy labels, SMALL and LARGE as shown in Figure 1. The values of variables with finite ranges were scaled to the range [0,1], while the value of path roughness was scaled so that the expected roughness would correspond to 0.5. The expected roughness was calculated as the expected value of each deformation multiplied by the number of deformations in the tile world and divided by the area of the world.

The learning rate for the k-th fuzzy state s_k is given by

$$\alpha_t^k = \frac{1}{\sum_{\tau=0}^{t-1} \mu_k(i_\tau)} \qquad (10)$$

where $\mu_k(i_t)$ is the degree of membership of a state i_t to fuzzy state s_k. The above formula is a natural extension of decreasing the learning rate for crisp states, where the learning rate for state i at time t is 1 divided by the number of visits to state i before time t.

At the beginning of every experiment, all Q-values were set to 0. We found that allowing agents to take a small fraction of random exploratory actions only decreased their performance, which is probably due to the fact that the stochastic nature of our world provided enough exploration as it is. Also, value function approximation architectures come with a benefit of automatic exploration, and therefore suboptimal actions exploring the environment are not needed. The value of a state-action pair (i, a) gets updated when the agent takes action a in any state j similar to state i. Therefore, taking action a in any state contributes to the Q-values of state-action pairs (i, a) for all states i via a chain reaction.

The issue of exploration can also be solved by initializing all Q-values higher than the actual ones. In this case, the value of any action will be decreased after it is chosen, and the competing actions will become preferable. This approach guarantees that the agent will explore all actions in the states it visits often. However, in this case it can take a very long time for the agent to settle onto the optimal policy, and that agent's performance at the early stages of learning will be very poor. In our problem, the true Q-values range between -50 and 50 depending on the fuzzy state. After comparing performance of an

agent initialized with Q-values of 50 with an agent initialized with Q-values equal to 0, we found that during the first 100 time steps the first agent obtained performance similar to a random agent (i.e., the one that randomly chooses between available opportunities), while the later agent consistently performed much better. Because of the above discussion, no exploration was done in the experiments described below.

Cooperation algorithms have the greatest influence on performance at the early stages of learning when agents have not finished exploring sufficiently the whole state space. Therefore, we used a very short time span of a few hundred time steps to compare performances of different algorithms.

According to the average reward framework, the performance of an agent is given by the sum of all rewards minus the sum of all step costs divided by the number of time steps. Performance of the multi-agent team was defined as the average of this measure over all agents. Note that this measure is equivalent to the total reward obtained during the whole simulation. We have also experimented with the regular discounted reward learning. However, we found that this form of learning implicitly teaches agents to maximize reward obtained per episode. Therefore, rather than choosing twice in a row an opportunity 10 steps away that has a Q-value of 10, the agent would choose to go for an opportunity 20 steps away that has a Q-value of 15. This is an undesirable behavior in our domain, and hence we chose the average reward framework for learning.

All performance results represent averages of 5000 simulations on the testing data. Such a large number of simulations was needed because of a highly stochastic nature of our testing domain. In each simulation, the training period lasted for a specified number of time steps, and the testing period lasted for 500 time steps.

A. Experimental Results and Discussion

We used MC learning with an initial learning rate $\alpha = 0.1$ for the first set of experiments. We found that for every parameter set the agent learns to rank correctly most of the 16 fuzzy states in less than 1000 episodes: the value of a state increases as the distance to the opportunity decreases, the opportunity reward increases, the mean lifetime of the opportunity increases, and the path roughness decreases. Examples of the state values after 1000 iterations for step cost scaling equal to 25 are shown in Table 1.

As Table 1 suggests, the learned fuzzy state values are dependent on the model parameters, such as the step cost scaling parameter, mean lifetime, etc. For example, as the step cost decreases or the mean lifetime of opportunities increases, the agent learns to go for the opportunity with

DISTANCE	REWARD	ROUGHNESS	LIFETIME	VALUE
LARGE1	LARGE2	LARGE3	LARGE4	-3.24
LARGE1	LARGE2	LARGE3	SMALL4	-1.41
LARGE1	LARGE2	SMALL3	LARGE4	-2.49
LARGE1	LARGE2	SMALL3	SMALL4	-1.88
LARGE1	SMALL2	LARGE3	LARGE4	-0.75
LARGE1	SMALL2	LARGE3	SMALL4	-2.25
LARGE1	SMALL2	SMALL3	LARGE4	-2.88
LARGE1	SMALL2	SMALL3	SMALL4	-0.67
SMALL1	LARGE2	LARGE3	LARGE4	-3.28
SMALL1	LARGE2	LARGE3	SMALL4	-1.50
SMALL1	LARGE2	SMALL3	LARGE4	22.74
SMALL1	LARGE2	SMALL3	SMALL4	3.62
SMALL1	SMALL2	LARGE3	LARGE4	-1.14
SMALL1	SMALL2	LARGE3	SMALL4	-1.26
SMALL1	SMALL2	SMALL3	LARGE4	-0.30
SMALL1	SMALL2	SMALL3	SMALL4	-1.82

Table 1: Examples of state values.

Team-optimal agents ($S = 0.5$)	$\mu = 5.40$	$\sigma = 0.12$
Selfish agents ($S = 1$)	$\mu = 6.02$	$\sigma = 0.08$
Team-conscious agents ($I = 0.3$)	$\mu = 6.18$	$\sigma = 0.06$

Table 2: Comparison of coordination methods

the highest value more often than for the closest one.

In our first set of experiments, we compare performance of 3 agents in a single world learning for 200 time steps while using different coordination strategies. The step cost scaling parameter was set to 5 for these experiments. The team's average reward and standard deviation over 5000 simulations are shown in Table 2.

In these experiments, team-optimal behavior does not result in a best performance because the coordination mechanism is designed to work when the Q-values of all agents have converged. Since our experiments test agents at the early stages of learning, it might not be wise to give up an opportunity to an agent that is far away but that wants it more because its value might erroneously be too high. In this case, the far away agent will incur too much cost while travelling to this opportunity and will reduce performance of the team. Therefore, team-conscious agents have a penalty that they impose on competing agents for being further away from an opportunity than themselves. More specifically, the parameter I=0.3 implies that an agent increases its selfishness parameter S by 0.3 for every unit of distance that the other agent is further away from the opportunity.

In the next set of experiments we compare effects of cooperative and independent learning when only one agent is present in each world. We used 3 agents in these exper-

3 cooperative agents, $t = 200$	$\mu = 7.63$	$\sigma = 0.09$
3 independent agents, $t = 600$	$\mu = 7.61$	$\sigma = 0.06$
3 non-learning agents	$\mu = 3.58$	$\sigma = 0.07$

Table 3: Comparison of cooperative and independent learning for SCS=5

3 cooperative agents, $t = 200$	$\mu = 6.32$	$\sigma = 0.09$
3 independent agents, $t = 600$	$\mu = 6.04$	$\sigma = 0.07$
3 non-learning agents	$\mu = 0.44$	$\sigma = 0.08$

Table 4: Comparison of cooperative and independent learning for SCS=25

3 independent agents	$\mu = 6.73$	$\sigma = 0.05$
3 cooperative agents	$\mu = 3.80$	$\sigma = 0.06$
3 parallel cooperative agents	$\mu = 6.68$	$\sigma = 0.07$

Table 5: Effects of coordination on cooperation for SCS=5

3 independent agents	$\mu = 3.52$	$\sigma = 0.09$
3 cooperative agents	$\mu = -0.48$	$\sigma = 0.08$
3 parallel cooperative agents	$\mu = 3.99$	$\sigma = 0.11$

Table 6: Effects of coordination on cooperation for SCS=25

iments, one per world. Cooperative agents were learning for 200 time steps, while independent agents were learning for 600 time steps. The step cost scaling parameter SCS was set to either 5 or 25, corresponding to low and high degrees of partial observability in the environment. As a benchmark, we also evaluated performance of non-learning agents that acted based on initial Q-values. These agents would randomly choose the opportunity to pursue, since all Q-values were initialized to be equal. The obtained results are summarized in Tables 3 and 4.

These results show that cooperative agents consistently outperform independent agents learning over a three times longer time interval. This result is very counterintuitive, since three agents sharing state values basically perform three times more exploration than a single agent during the same time interval. It is also interesting to note that the performance improvement of cooperative over independent agents increases with an increasing level of partial observability present in the environment.

Finally, we conduct experiments showing the effects of coordination on cooperation. We first compare performance of 3 independent agents coordinating in the same world with 3 cooperative agents coordinating in the same world. The last experiment considers two parallel worlds with 3 agents coordinating in each world. In this experiment, agent i in world 1 cooperates with agent i in world 2. In these experiments, independent agents were learning for 400 time steps while cooperative agents were learning for 200 time steps. The results are summarized in Tables 5 and 6.

This experiment shows the unexpected results that cooperative agents that are learning in a competitive environment perform much worse than independent agents learning in the same environment. This might be explained by the fact that experience sharing reduces di-

versity between agents and hence increases the number of conflicts they have when choosing the opportunity to pursue. This hypothesis is supported by the superior performance of cooperative agents in the third setup of the experiments, where agents were cooperating with their twins in a parallel world while maintaining diversity within the same world. Also, just as in previous experiments, for a low degree of partial observability the cooperative performance is comparable to independent performance over a twice longer time interval. As the partial observability is increased, cooperation shows clear advantages over independent performance.

A very useful implication of the above result is demonstrated in the last set of experiments. In the first setup, a 20-by-40 world is populated by 2 agents, which learn independently. In the second setup, 2 agents learn cooperatively in the same world. In the third setup, the world is divided in two parts, 20-by-20 each. The two agents learn independently, and each agent is able to see only those opportunities that appear in its half. The fourth setup is just like the third one, except that agents are allowed to cooperate. All agents learned for 200 time steps. The results for the step cost scaling parameter equal to 5 are given in Table 7.

The results show that performance of several agents competing for the same set of opportunities can be in-

2 independent agents, 20x40 world	$\mu = 3.23$	$\sigma = 0.05$
2 cooperative agents, 20x40 world	$\mu = 1.61$	$\sigma = 0.09$
2 independent agents, split world	$\mu = 5.48$	$\sigma = 0.08$
2 cooperative agents, split world	$\mu = 6.24$	$\sigma = 0.10$

Table 7: Taking full advantage of cooperation by eliminating the coordination problem.

creased by restricting the agents' operations to separate regions of the world. Besides the basic advantages of this divide-and-conquer approach, spatial specialization also allows agents to take a full advantage of cooperation through experience sharing, since the coordination problem gets eliminated.

VI. CONCLUSIONS

We showed that a team-optimal coordination strategy does not have to give team optimal results, which should alert other researchers to the issue of multi-agent coordination. We have also tested how conclusions obtained by other researchers about performance of cooperation algorithms in small MDP's carry over to POMDP's with continuous state spaces.

We found that the cooperation method of common policy updating performs very well in partially observable environments and that its benefits increase with an increasing level of partial observability present in the environment. In fact, we showed that K agents learning cooperatively over N time steps significantly outperform K independent agents learning over K*N time steps, contrary to the theoretical results obtained for MDP's [Whitehead, 1991].

Finally, we have shown that a tradeoff exists between experience sharing and coordination. One of the implications of this tradeoff is that introducing spatial restrictions on operations of autonomous agents in competitive environments can actually improve the team performance by eliminating the coordination problem and taking full advantages of cooperative learning.

REFERENCES

Berenji, H. R. 1997. Collaborating fuzzy reinforcement learning agents. In Proceedings of the International Fuzzy Systems Associations Conference (IFSA '97), Prague, Czech Republic.

Kaelbling, L. P.; Littman, M. L.; and Andrew W. Moore. 1996. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, Volume 4.

Kelly, I. D. and Keating, D. A. 1998. Increased learning rates through the sharing of experiences of multiple autonomous mobile robot agents. In proceedings of FUZZ-IEEE '98. Anchorage, Alaska.

Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In Proceedings of the Eleventh International Conference on Machine Learning, 157-163. San Francisco, CA: Morgan Kaufmann.

Luck, M. 1997. Foundations of multi-agent systems: issues and Directions. The Knowledge Engineering Review, 12(3): 307-308.

Mataric, M. J. 1997. Reinforcement Learning in the Multi-Robot Domain, Autonomous Robots, 4(1).

Stone, P. and Veloso, M. 1997. Multiagent systems: A survey from a machine learning perspective. Technical Report, CMU-CS-97-193, Dept. of Computer Science, Carnegie Mellon Univ.

Stone, P. and Veloso, M. 1999. Team-Partitioned, Opaque-Transition Reinforcement Learning, "In Proceedings of Third International Conference on Autonomous Agents (Agents'99)".

Sutton, R.S., and Barto, A.G. 1998. Reinforcement Learning: An Introduction. MIT Press.

Sycara, K. P. 1998. Multiagent systems. AI Magazine, summer 1998, 79-92.

Sycara, K.; Decker, K.; Pannu, A.; Williamson, M.; and Zeng, D. 1996. Distributed intelligent agents. IEEE Expert 11(6): 36-46.

Tan, M. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In Proceedings of the Tenth International Conference on Machine Learning, 330-337.

Whitehead, S. D. (1991) A complexity analysis of cooperative mechanisms in reinforcement learning. In Proceedings of AAAI-91, pp. 607-613.