LEARNING TO COORDINATE IN TOPOLOGICAL NAVIGATION TASKS [1]

Francisco S. Melo [*,2] **Isabel Ribeiro** [*]

[*] *Institute for Systems and Robotics*
Instituto Superior Técnico
Av. Rovisco Pais, 1
1049-001 Lisboa,
PORTUGAL

Abstract: In this paper we address multi-robot coordinated navigation from a topological perspective. The adopted topological representation of the environment leads naturally to a Markov game model to describe the interaction of the multiple robots in the environment. In this setting, we combine the Q-learning algorithm with a powerful coordination mechanism (biased adaptive play). We show that this combined algorithm, *coordinated Q-learning*, converges to an optimal, coordinated solution for the navigation problem. This implies that the team of robots is able to coordinate without using any communication protocol to enforce the coordination. We illustrate our method in some simple navigation tasks.

Keywords: Topological navigation, Markov games, Biased adaptive play, Coordination

1. INTRODUCTION

In recent years, particular interest has been devoted to the use of topological maps in robotic navigation. A topological map represents an environment as a discrete set of states (the nodes in a graph) and the transition information between the states (the edges of a graph). This type of environments may be easily described using Markov processes and, in fact, Markov processes have already been used to model robotic navigation tasks using different methodologies, (Cassandra et al., 1987; Melo and Ribeiro, 2005).

In this paper, we are interested in addressing *cooperative multi-robot navigation problems.* Research on cooperative multi-robot systems typically focuses on three fundamental issues (Cao et al., 1997): the *task* to accomplish, the *mechanism of cooperation* and the attained *performance.* The model of *team Markov games*, used in this paper, immediately settles two of the fundamental issues referred above, by considering a reward structure that, simultaneously, *defines the task* and is used to *evaluate the performance* of the team.

We consider problems in which a group of robots must navigate from an initial configuration to a final configuration, avoiding "accidents" resulting from mis-coordinations. To this purpose, all robots must coordinate in a common "strategy". We admit, however, that no explicit communication takes place, *i.e.*, consensus in this common joint strategy must *emerge* from the mutual interaction among the different robots and with the environment. [3] Therefore, the class of problems

[1] This work was partially supported by Programa Operacional Sociedade do Conhecimento (POS_C) that includes FEDER funds.

[2] The author acknowledges the PhD grant SFRH/BD/3074/2000.

[3] The consideration of no explicit communication can be supported by several arguments. We do not pursue such

considered herein feature cooperation as *coordination*: the multiple decision-makers must *coordinate* their individual decisions to yield an optimal joint behavior. Further references on coordination in navigation tasks include Alami et al. (1994); Bennewitz et al. (2001); LaValle and Hutchinson (1996) and references therein.

We use a Markov game with identical interests to model a group of mobile robots that must perform some navigation task in a coordinated fashion (such as reaching a goal configuration). We propose a learning algorithm that ensures coordination without relying on any explicit communication between the different robots.

2. TOPOLOGICAL NAVIGATION AND MARKOV MODELS

In this section we introduce the framework of Markov models used throughout the paper.

2.1 Markov decision processes

Consider a mobile robot moving in an environment. Suppose that such robot must perform some given task in a specific location of the environment. Suppose, furthermore, that the environment may be represented by a topological map, as the one depicted in Fig. 1.

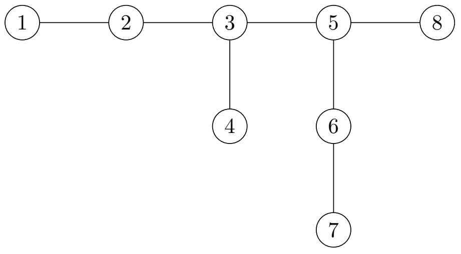

Fig. 1. Example of a topological map.

In a topological map, the environment is discretized in a set $\mathcal{X}$ of *states* corresponding to possible "topological locations" for the robot. At each time instant t, the robot has available a finite set $\mathcal{A}$ of possible actions (*e.g.*, "move North", "go to state 2", etc.). Whenever the robot chooses a give action $a \in \mathcal{A}$ in state $i \in \mathcal{X}$, it will move to state $j \in \mathcal{X}$ with probability $\mathbb{P}\left[X_{t+1} = j \mid X_t = i, A_t = a\right] = \mathsf{P}_a(i,j)$, where X_t and A_t represent the state and the action of the robot at time instant t. For each $a \in \mathcal{A}$, P_a is the *transition probability matrix.*

We consider that every time the robot moves from state i to state j by taking an action a, it is rewarded with a numerical reward $r(i,a,j)$. This *reinforcement signal* provides the robot with *evaluative feedback*: the reward works as a quantitative evaluation on "how well the robot is doing" and formalizes the navigation purpose for the robot. The robot must then choose the sequence of control actions, $\{A_t\}$, so as to maximize the functional

$$V(i,\{A_t\}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(X_t, A_t) \mid X_t = i\right], \qquad (1)$$

where $R(X_t, A_t)$ is the *random* reward received at time instant t and $0 < \gamma < 1$ is a discount factor assigning greater importance to more immediate rewards than to those in a distant future. Notice that $R(x,a)$ in (1) is a random variable, since it depends on the (random) state transition. This occurs even if r is deterministic. The tuple $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ is a *Markov decision process* (MDP) and is a suitable model for single-robot topological navigation tasks.

Given an MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, the *optimal value function* V^* is defined for each state $i \in \mathcal{X}$ as

$$V^*(i) = \max_{\{A_t\}} \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^t R(X_t, A_t) \mid X_0 = i\right] \qquad (2)$$

and verifies

$$V^*(i) = \max_{a\in\mathcal{A}} \sum_{j\in\mathcal{X}} \mathsf{P}_a(i,j)\big[r(i,a,j) + \gamma V^*(j)\big], \qquad (3)$$

which is a form of the Bellman optimality equation. The optimal Q-values $Q^*(i,a)$ are defined for each state-action pair $(i,a) \in \mathcal{X} \times \mathcal{A}$ as

$$Q^*(i,a) = \sum_{j\in\mathcal{X}} \mathsf{P}_a(i,j)\big[r(i,a,j) + \gamma V^*(j)\big]. \qquad (4)$$

If $V^*(i)$ "measures" the total discounted reward obtained during an expectedly optimal trajectory starting at state i, $Q^*(i,a)$ measures the total discounted reward obtained during an expectedly optimal trajectory starting at state i when the first action is a. We notice that the optimal control sequence $\{A_t\}$ can be easily computed from Q^* as a function of the state, $A_t(X_t) = \max_{a\in\mathcal{A}} Q^*(X_t, a)$, so we can focus the learning process on determining Q^*.

The optimal Q-function can be approximated by a sequence of functions $\{Q_n\}$, generated recursively by

$$\begin{aligned} Q_{n+1}(i,a) = Q_n(i,a) + \alpha_n(i,a)\big[R(i,a) + \\ + \gamma \max_{b\in\mathcal{A}} Q_n(X(i,a),b) - Q_n(i,a)\big], \end{aligned} \qquad (5)$$

where $X(i,a)$ is random variable obtained according to the transition probabilities defined by P and $\{\alpha_n(i,a)\}$ is a sequence of step-sizes. The sequence $\{Q_n\}$ will converge to Q^* as long as each pair (i,a) is "visited" infinitely often and the step-sizes are adequately chosen (Watkins, 1989). Expression (5) is the update equation of *Q-learning*, a widely known method that we use in our multi-robot algorithm.

2.2 Team Markov games

Markov games can be interpreted as generalizations of MDPs to multiple robots. Therefore, a

argument here and refer to several works that discuss these issues in greater detail (Durfee et al., 1987; Tsitsiklis and Athans, 1985).

Markov game is a tuple $\left(N, \mathcal{X}, (\mathcal{A}^k), \mathsf{P}, (r^k), \gamma\right)$, where N is the number of players, $\mathcal{X}$ is the state-space, $\mathcal{A} = \times_{k=1}^N \mathcal{A}^k$ is the set of joint actions, P is the controlled transition matrix and r^k is the reward function for player k.

In this paper, we are interested in *fully co-operative* Markov games, also known as *team Markov games*. A team Markov game is a tuple $\left(N, \mathcal{X}, (\mathcal{A}^k), \mathsf{P}, r, \gamma\right)$ where the reward function r is common to all players.

In our model all robots share the same goal, which is to maximize the total expected reward over all admissible control sequences $\{A_t\}$, defined as in (1), where now $R(i, a)$ is the random reward received by *all* robots for taking the joint action a in state i. It is immediate to define the *optimal value function* V^* for a team Markov game as in (2), where now A_t stands for the joint action at time t. This optimal value function also verifies (3) and we can define the optimal Q-function, Q^*, as in (4).

To further clarify the relation between MDPs and team Markov games, consider a group of robots that must complete some navigation task (*e.g.*, reach some pre-specified configuration). Markov decision processes describe this situation *when the control of the robots is centralized*. Team Markov games describe this exact same situation but *when the control of the robots is decentralized*, *i.e.*, each robot chooses its own action independently of the other robots.

3. COORDINATION

Consider the scenario depicted in Fig. 2.

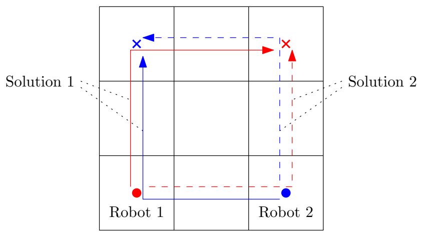

Fig. 2. Example with 2 robots in a 2×2 grid-world.

Two robots (1 and 2) must move from the corresponding cell in the bottom row to the opposite cell in the top row, without colliding with each other (*i.e.*, lying in the same cell). There are several optimal ways of doing this, two of which are depicted in Fig. 2. Suppose now that Robot 1 opts by choosing Solution 2 and Robot 2 opts by choosing Solution 1 (recall that we assume no communication). This means that they will collide in the middle cell in the bottom row, which is an undesirable behavior.

This problem is known as a *coordination problem* (Boutilier, 1999). Even if all robots know the model and the solutions, it is still necessary to devise some specific mechanism to ensure that, in the presence of multiple solutions, all robots will commit to the same one. This mechanism can rely on implicit assumptions on the way the other robots choose their actions (Lauer and Riedmiller, 2000), communication (Fischer et al., 2004), social conventions (Findler and Malyankar, 2000) or coordination graphs (Guestrin et al., 2002).

In this paper we are interested in addressing coordination as a result of interaction among the robots: coordination should *emerge* from the interaction among the several robots rather than being “intrinsically implanted”. We also consider that no *explicit communication* takes place.

Several works in the literature address the problem of emerging coordination in multi-agent systems, all relying in history-based coordination mechanisms, such as *fictitious play* (Claus and Boutilier, 1998), *adaptive play* (Young, 1993) or *biased adaptive play* (Wang and Sandholm, 2003). In this paper we use biased adaptive play (BAP), since it actually provides stronger convergence guarantees and combine it with Q-learning.

This combined method, dubbed as *coordinated Q-learning* (CQL), endows the robots with a powerful way of *learning how to complete the task* using no model of the world while simultaneously *learning to coordinate in the execution of that task.* We further mention that, in Wang and Sandholm (2003), BAP is combined with a simplified variant of *adaptive real-time dynamic programming* (ARTDP), a model-based learning algorithm. In that same work, the authors mention the interest of combining biased adaptive play with a model-free learning algorithm, contribution that is provided in our paper.

3.1 Biased adaptive play

We now briefly describe BAP and its main properties. Due to space limitations, we do not dwell on the details of the algorithm and refer to Melo and Ribeiro (2007) for a more thorough treatment.

To describe how BAP works, we start by remarking that coordination at each state $i \in \mathcal{X}$ can be attained by considering the team matrix game $\Gamma_i = (N, (\mathcal{A}^k), Q^*(i, \cdot))$. This matrix game consists of a set of N players, each with a set $\mathcal{A}^k$ of available actions. In this class of games, all players “receive a payoff” of $Q^*(i, a)$ whenever they play the joint action a. This is a *one-shot game* where every player wants to maximize the attained reward in a single play. Coordination in the original team Markov game can be attained by coordinating in each of the matrix games Γ_i (Boutilier, 1999). This is the purpose of BAP.

To describe BAP, we first construct a *virtual game* $VG(i) = \left(N, (\mathcal{A}^k), r_{VG}\right)$ from Γ_i, where $r_{VG}(a) = 1$ if $a = \mathbf{opt}(i)$ and 0 otherwise. We denoted by $\mathbf{opt}(i)$ the set of optimal joint actions in state i. We allow the N players to repeatedly engage in this virtual game and let H_t be a vector with the last m joint plays at the t^{th} play of the game. Now given two integers $K \leq m$, we refer to a set $K(H_t)$ of K samples randomly drawn from H_t without replacement as a *K-sample*. A "BAP player" k draws a K-sample from the history of the m most recent plays and checks if

(1) There is a joint action $a^* \in \mathbf{opt}(i)$ such that, for all the actions $a \in K(H_t)$, $a^{-k} = (a^*)^{-k}$;
(2) There is at least one action $a^* \in \mathbf{opt}(i)$ such that $a^* \in K(H_t)$.

If these two conditions are verified, player k believes that the remaining players have coordinated in an optimal action $(a^*)^{-k}$. and chooses the last played action $a^* \in \mathbf{opt}(i)$ verifying 2. If either 1 or 2 (or both) do not hold, player k uses the K-sample to estimate the strategies of the other players and chooses its action as a best response to this estimate. It has been shown that BAP ensures coordination w.p.1 as $t \to \infty$ as long as $m \geq K(N+2)$—see Theorems 1 and 3 and Lemma 4 in (Wang and Sandholm, 2003).

4. COORDINATED Q-LEARNING

In our description of the CQL algorithm, we consider two essential components: (1) *learning the game*; and (2) *learning to coordinate*. Learning the game consists in estimating the optimal Q-function; learning to coordinate consists in agreeing upon an optimal joint behavior.

4.1 Learning the game

In CQL, each robot uses the Q-learning update rule in (5) to learn the optimal Q-values. Since all robots receive the same reward and admittedly observe (a posteriori) all actions chosen, all robots maintain, at each time instant, a common estimate Q_t of Q^*. As seen in Section 2, if every state-action pair (i, a) is visited infinitely often, the sequence $\{Q_t\}$ converges to Q^* with probability 1, *independently of the strategy used to explore the environment*. This result is standard and can be found in numerous references in the literature, *e.g.*, Littman (2001).

On the other hand, it is important to ensure sufficient *exploration* as the robots coordinate. In fact, it is important to ensure sufficient visits to every pair (i, a), even as the robots coordinate. The use of exploration policies that become *greedy in the limit* has been shown to settle this issue in a satisfactory way, for the purposes of this paper. Such policies, known as *greedy in the limit with infinite exploration* (GLIE) were thoroughly studied in Singh et al. (2000) and their application to multi-agent scenarios in Littman (2001); Wang and Sandholm (2003). In this work, we adopt a Boltzmann-greedy policy that explores with a probability given by a Boltzmann distribution.

4.2 Learning to coordinate

Recall that, in BAP, the players use the function Q^* to define a matrix game Γ_i to ensure coordination. However, during learning, the robots do not know the function Q^*, but rather an approximate estimate Q_t of Q^* that they must use to coordinate.

To achieve coordination during learning we consider a sequence $\{VG_t\}$ of virtual games built from the estimates Q_t by making use of the concept of ε-optimal actions. Each virtual game VG_t is thus defined as a tuple $VG_t = \big(N, (\mathcal{A}^k), r_t\big)$, where the payoff function is

$$r_t(a) = \begin{cases} 1 & \text{if } a \in \mathbf{opt}^{\varepsilon_t}(i); \\ 0 & \text{otherwise.} \end{cases}$$

We denoted by $\mathbf{opt}^{\varepsilon_t}(i)$ the set of ε_t-optimal actions with respect to Q_t at state i. As $\varepsilon_t \to 0$, all suboptimal actions are eliminated from the virtual games VG_t and we need only guarantee that $\varepsilon_t \to 0$ more slowly than $Q_t \to Q^*$. This is established in the following result.

Theorem 1. Let $\Gamma = \big(N, \mathcal{X}, (\mathcal{A}^k), \mathsf{P}, r, \gamma\big)$ be a team Markov game with N players. Suppose that the following conditions hold:

(1) The players use the Q-learning update rule in (5) to learn the optimal Q-function;
(2) The players use BAP with GLIE exploration to coordinate in each stage-game;
(3) Each virtual game VG_t used in BAP considers ε_t-optimal strategies;
(4) The sequence $\{\varepsilon_t\}$ decreases monotonically to zero and verifies
$$\lim_{t\to\infty} \frac{\sqrt{\frac{\log\log(N_t)}{N_t}}}{\varepsilon_t} = 0, \tag{6}$$
where N_t is number of visits to the least visited state-action pair at time t;
(5) The the lengths of the history H_t and K-sample h, m and K, verify $m \geq K(N+2)$.

Then, the sequence of estimates $\{Q_t^k\}$ generated by CQL converges to Q^* w.p.1. Furthermore, all players in N coordinate in an optimal Nash equilibrium of Γ w.p.1.

PROOF See Melo and Ribeiro (2007) □

5. RESULTS

We now present the results obtained in several simple sample scenarios using the CQL algorithm.

The first set of results was obtained in several small grid-world problems. We present in Figures 4 and 5 the results obtained in 2×2 and 3×3 grid worlds. In both problems, each of two players must reach the opposite corner (see Fig. 3).

When both players reach the corresponding corners, they receive a common reward of 20. If they "collide" in some state, they receive a reward of -10. Otherwise, they receive a reward of 0. The robots can move in one of four directions: N, S, E and W, these being the 4 available individual actions. When a robot chooses to move in a given direction, it moves to the adjacent state in that direction (if there is one) with probability 0.9 and remains in the same state with probability 0.1. Since we consider only two robots moving, each with 4 possible actions, this makes a total of 16 possible joint actions. The number of states is 16 and 81, respectively.

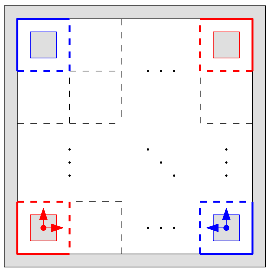

Fig. 3. Generic $n \times n$ grid world.

In each of the Figures 4 and 5 we presented the total reward obtained during learning and the probability of coordination/exploration. Initially, the robots explored the environment (low coordination probability). As the probability of coordination approaches 1, the robots start coordinating in their action choice, as seen by the inflection in the reward plots. This means that in both grid-world tests both players learned to coordinate using the CQL algorithm. Notice that the probability of coordination converges to 1 and, as seen in the cumulative reward plot, they are capable of avoiding collision (both curves present positive slope). Notice that coordination in the 2×2 grid is "harder" than in the 3×3 grid, since there is less space to avoid collisions.

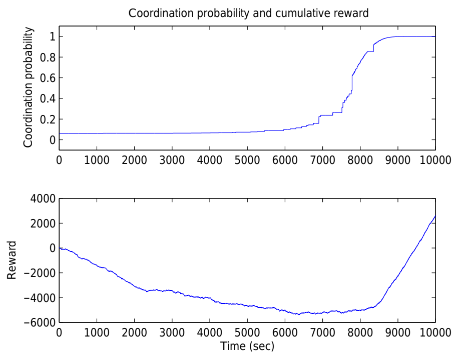

Fig. 4. Cumulative reward and probability of coordination obtained in the 2×2 navigation problem.

The second test considers a somewhat more elaborate scenario, depicted in Fig. 6. In this scenario,

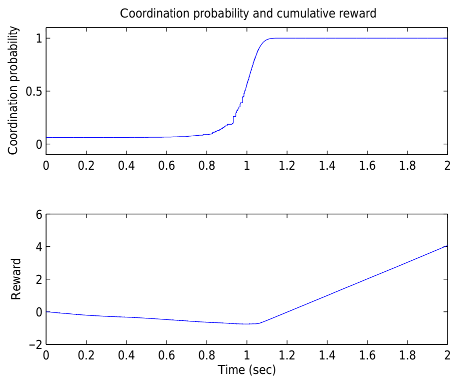

Fig. 5. Cumulative reward and probability of coordination obtained in the 3×3 grid navigation problem. The values in the time scale should be multiplied by 10^4.

the robots can be in one of 4 possible orientations in each cell (N, S, E and W) and 3 possible actions: turn left, turn right and move forward. The actions turn-left/turn-right make a 90° turn in the corresponding direction with a probability of 0.9. With a probability of 0.05 the robots either keep the same orientation unchanged, or turn 180°. The move action behaves similarly to the actions in the grid-world.

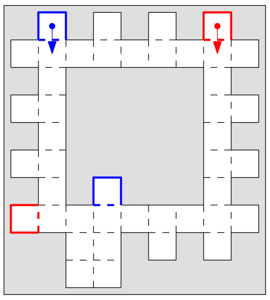

Fig. 6. Small maze world.

Notice that, although the number of joint actions is now only 9, the total number of states is 29 584. We also assumed that the robots have some uncertainty regarding their joint position in the environment, arising from sensor noise. They keep, at each step, a "belief" vector, describing the probability of being in each of the possible joint states. The robots then decide according to the most likely state. The obtained performance is depicted in Fig. 7.

6. CONCLUDING REMARKS

We conclude the paper with several important remarks. First of all, the CQL algorithm is closely related to optimal adaptive learning (OAL) as described in Wang and Sandholm (2003). While CQL combines Q-learning with biased adaptive play, OAL combines model-based learning with biased adaptive play. As detailed in Melo and Ribeiro (2007), the only complication in combining Q-learning with biased adaptive play resides

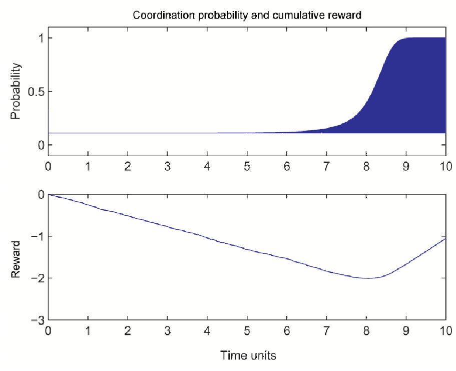

Fig. 7. Cumulative reward and probability of coordination obtained in the larger navigation problem. The values in both scales should be multiplied by 10^5.

in suitably choosing a decay rate for the sequence ε_t.

A second remark is related with the problem of coordination in multi-robot learning problems. When considering multi-robot learning problems, coordination should always be explicitly accounted for. The existence of multiple optimal strategies may lead the joint behavior of a group of robots to be arbitrarily poor if no coordination is enforced, even if all robots know exactly the world and its model. If the robots are to learn to coordinate while learning the game itself, the coordination mechanism must be supported by the past history of the game.

Future work addressing complex environments (with large/infinite state-spaces) should take into account the impact of compact representations of the state-space on how coordination can now be obtained from the history of the process.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the helpful discussions with Prof. Manuela Veloso.

REFERENCES

R. Alami, F. Robert, F. Ingrand, and S. Suzuki. A paradigm for plan-merging and its use for multi-robot cooperation. In *Proc. 1994 IEEE Int. Conf. Systems, Man, and Cybernetics*, pages 612–617, 1994.

M. Bennewitz, W. Burgard, and S. Thrun. Optimizing schedules for prioritized path planning of multi-robot systems. In *Proc. 2001 IEEE Int. Conf. Robotics and Automation*, pages 271–276, 2001.

C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proc. 16th IJCAI*, pages 478–485, 1999.

Y. Cao, A. Fukunaga, and A. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4(1):1–23, 1997.

A. Cassandra, L. Kaelbling, and J. Kurien. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. *Math. Operations Research*, 12(3):441–450, 1987.

C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proc. 15th AAAI*, pages 746–752, 1998.

E. Durfee, V. Lesser, and D. Corkill. Coherent cooperation among communicating problem solvers. *IEEE Trans. Computers*, 36(11):1275–1291, 1987.

N. Findler and R. Malyankar. Social structures and the problem of coordination in intelligent agent societies. Invited talk at IMACS World Congress (2000), 2000.

F. Fischer, M. Rovatsos, and Gerhard Weiss. Hierarchical reinforcement learning in communication-mediated multiagent coordination. In *Proc. 3rd AAMAS*, pages 1334–1335, 2004.

C. Guestrin, M. Lagoudakis, and R. Parr. Coordinated reinforcement learning. In *Proc. 19th ICML*, pages 227–234, 2002.

M. Lauer and M. Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proc. 17th ICML*, pages 535–542, 2000.

Steven M. LaValle and Seth A. Hutchinson. Optimal motion planning for multiple robots having independent goals. In *Proc. 1996 IEEE Int. Conf. Robotics and Automation*, pages 2847–2852, 1996.

M. Littman. Value-function reinforcement learning in Markov games. *J. Cognitive Systems Research*, 2(1):55–66, 2001.

F. Melo and I. Ribeiro. Transition entropy in partially observable Markov decision processes. In *Proc. IAS-9*, 2005. (to appear).

F. Melo and I. Ribeiro. Rational and convergent model-free adaptive learning for team Markov games. Technical Report RT-601-07, Institute for Systems and Robotics, Feb. 2007.

S. Singh, T. Jaakkola, M. Littman, and C. Szepesvari. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3):287–310, 2000.

J. Tsitsiklis and M. Athans. On the complexity of decentralized decision making and detection problems. *IEEE Trans. Automatic Control*, AC-30(5):440–446, 1985.

X. Wang and T. Sandholm. Reinforcement learning to play an optimal Nash equilibrium in team Markov games. In *Procs. NIPS'03*, pages 1571–1578. 2003.

C. Watkins. *Learning from delayed rewards*. PhD thesis, King's College, Cambridge Univ., May 1989.

H. Peyton Young. The evolution of conventions. *Econometrica*, 61(1):57–84, 1993.