Q-learning in Two-Player Two-Action Games

Monica Babes
Rutgers University
babes@cs.rutgers.edu

Michael Wunder
Rutgers University
mwunder@cs.rutgers.edu

Michael Littman
Rutgers University
mlittman@cs.rutgers.edu

ABSTRACT

Q-learning is a simple, powerful algorithm for behavior learning. It was derived in the context of single agent decision making in Markov decision process environments, but its applicability is much more broad—in experiments in multiagent environments, Q-learning has also performed well. Our preliminary analysis finds that Q-learning's indirect control of behavior via estimates of value contributes to its beneficial performance in general-sum 2-player games like the Prisoner's Dilemma.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Algorithms—Intelligent Agents,Multiagent Systems

Keywords

Reinforcement learning, Multiagent learning, dynamical systems

1. INTRODUCTION

Q-learning [20] was one of the first algorithms for reinforcement learning [18] specifically designed to maximize reward in multistage environments. Several authors have shown that it converges to optimal values [21, 19, 8, 11] and optimal behavior [15] in Markov decision process environments. Variants have even been shown to converge to near optimal behavior in polynomial time [17].

General sum games provide a formal framework for defining and addressing important problems in agent interaction. Given the importance of learning in multiagent settings and the success of Q-learning in single agent environments, it is unsurprising that researchers have applied Q-learning here as well. There are many positive empirical results in which Q-learning performs well, sometimes even beyond expectations [14, 12, 10, 3, 2, 13].

Positive theoretical results have been generalized to some special multiagent environments, in particular when the reward structures are strictly cooperative or competitive [9]. Although Q-learning variants have been defined for general sum games [7, 6], the very idea of representing multiagent strategies using Q values has been shown to extremely problematic [23]. Thus, positive theoretical results have been lacking.

This state of affairs leads us to an enigma. Why is it the Q-learning performs well in multiagent environments even though existing analyses don't support its use in this setting? We don't know. But, in this paper, we start collecting clues in the setting of 2-agent, 2-action games. The majority of our results are presented in the context of the Iterated Prisoner's dilemma or IPD [1] because of its simplicity as well as its wide applicability. In addition, its deceptive nature makes it an interesting testbed. In Section 2, we demonstrate Q-learning performing well in this environment. In Section 3, we shed some light on the observed behavior by casting Q-learning into a dynamical system framework and showing that high scoring policies are stable, in some sense. In Section 4, we attack the problem from another angle and cast Q-learning's dynamics as a probabilistic process, showing that the transition between cooperative and defecting behavior can be characterized, at least partially. We conclude with suggestions for how these insights could be exploited in the design or analysis of future multiagent learning algorithms.

1.1 Definitions

A game is a description of the interaction between two or more agents. It typically specifies the number of agents involved in the interaction—two, in our case—and the size of the action set A—again, two, here. The action set includes all the actions that are available to the participating agents and the reward functions R_i for agent i map the set of joint actions of all the agents to the payoff to agent i.

In the iterated or *repeated games* we study, at each step the environment dispenses rewards to each agent according to the joint action taken by all the agents in the game. A *best response* is a strategy that chooses the action that maximizes the agent's payoff given a fixed specification of the action policies of all the other agents.

A *Nash equilibrium* is a joint strategy in which each player's action is a best response to the strategies of its opponents. Although Nash equilibrium algorithms have been used for understanding and implementing multiagent systems, the Nash equilibrium concept is difficult to apply to the design of an individual agent.

Learning algorithms provide a sensible approach for focusing on defining agent behavior in a game from the perspective of a single agent. Many learning algorithms target the generation of a best response, so that no matter what behavior the other agent adopts, the learning agent will be

Cite as: Q-learning in Two-Player Two-Action Games, Wunder, Littman and Babes, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

striving to maximize its reward.

1.2 Q-learning

Q-learning is a single-agent learning algorithm that has been used in the multiagent setting. The Q-learner maintains a Q table, which is a data structure that stores a value for each state–action pair. In each state, for each action, the state–action (or Q) value represents the expected payoff that the agent receives from choosing the given action from that state, then selecting actions in future states to maximize expected payoff.

In a repeated game, the Q table for each Q-learning agent i consists of a vector of values, Q_i^a, with one component for each action $a \in A$. The Q-learning rule, simplified for this setting, can be written

$$Q_i^a \leftarrow Q_i^a + \alpha(r + \gamma \max_{a' \in A} Q_i^{a'} - Q_i^a),$$

where α is a learning rate or step-size parameter, $0 \leq \gamma < 1$ is a discount factor weighting future rewards relative to current rewards, and r is the payoff value received in response to action a.

Q-learning can be viewed as a family of algorithms that differ in their answers to the following questions:

1. How is Q_i^a initialized?
2. How is a chosen for the current timestep?
3. How is α selected for the current timestep?

In single-agent environments, it is known that convergence of Q_i^a to the value obtained for the optimal strategy follows as long as (1) initial values are finite, (2) all actions are chosen infinitely often, and (3) learning rates decay according to the rules of stochastic approximation theory. Against a fixed memoryless opponent, therefore, Q-learning converges to a best response. Against a dynamic opponent, such as another Q-learner, Q-learning need not converge.

Throughout this paper, we use ϵ-greedy action choices to answer Question 2. The ϵ-greedy action is the action a for which Q_i^a is maximized or, with probability ϵ, a randomly chosen action. Generally, reinforcement-learning researchers choose a relatively small value of ϵ so that most of the time the agent makes the choice that is best with respect to its learned values, but it still continues to try other actions so that their values will be estimated as well.

WoLF-IGA [4] (Win or Learn Fast using Infinitesimal Gradient Ascent) is an algorithm that is guaranteed to always converge to a stationary policy and to be rational; that is, to converge to a best-response strategy against opponents that converge to stationary policies. An important fact about WoLF-IGA is that it converges to a Nash equilibrium when playing a copy of itself (self play).

2. Q-LEARNING'S BEHAVIOR

The observation that motivated this work is that Q-learning behaves oddly in the classical Prisoner's dilemma game. In the Prisoner's dilemma (PD), two agents can both choose to cooperate (3 points) or both choose to defect (1 point). Given a fixed choice for the opponent, defecting is worth one point more than cooperating. So, an agent that defects on its opponent gets the "temptation" payoff of 4. An agent that cooperates when the opponent defects gets the "sucker" payoff of 0.

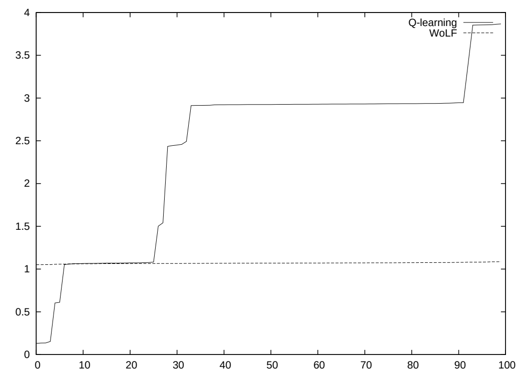

Figure 1: Distribution of outcomes obtained by two Q-learners and by two WoLF learners in IPD.

Since the agent's payoff is always better for defecting, regardless of the opponent's choice, mutual defection (DD) is the only Nash equilibrium in this game. Of course, the "dilemma" aspect of the game is that the resulting payoff of 1 for each player is worse than what they would get if they resisted making a best response and received the mutual cooperation (CC) score of 3 for each player. In the iterated Prisoner's dilemma (IPD), players face off in PD repeatedly.

Figure 1 illustrates the results of learning using WoLF-IGA and Q-learning agents in the IPD. We ran each agent in self play for 100 runs. Q-learning parameters were $\gamma = 0.9$, $\epsilon = 0.07$, $\alpha_q = (1 - 1/2000)^t$ where t is the number of steps since run started. For WoLF $\alpha_w = 1/(1 + n/500)$, where n is the number of steps in which the current action has been visited during the current run, and $\delta_w = 1/(1000 + n/10)$, $\delta_l = 4\delta_w$. Each run consisted of 300,000 steps of IPD. The average rewards were computed over the last 5000 steps, then the average reward of each run was recorded. We sorted all the average rewards we got for each pair of agents and generated one point for the average reward of each run. This way, we got a plot for the average rewards of each pair of players. We plotted them together to illustrate the difference between the performances of the two agents in self-play.

WoLF-IGA has been shown to converge to a Nash equilibrium, so it is not surprising that all runs end with WoLF-IGA obtaining approximately 1, the DD payoff.

More surprisingly is the outcome of Q-learning. Although many runs (about 1/4) end in mutual defection, a substantial number of runs (about 2/3) terminate with scores close to mutual cooperation. If Q-learning is driven to best responses and the only mutual best response in this game is DD, why does Q-learning do even better?

3. DYNAMICAL SYSTEMS APPROACH

IGA (Infinitesimal Gradient Descent), which gave rise to WoLF-IGA, was proposed as an abstract algorithm for 2-player, 2-action games [16]. The algorithm maintains an explicit policy for both players, which can be summarized as a single probability for each player specifying its chance of choosing the first action. These policies are updated by taking an infinitesimal step in the direction of the gradient—each player modifies its choices to maximize its expected reward.

Using a dynamical systems analysis, the authors showed that IGA players converge to a Nash equilibrium or to an

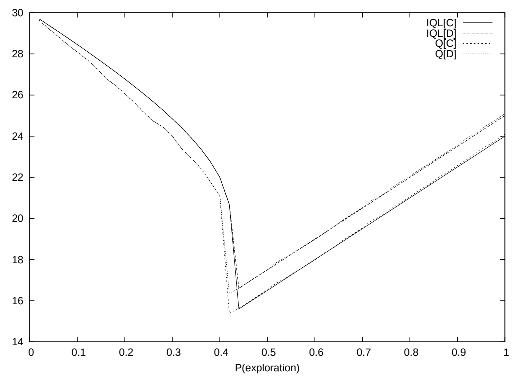

Figure 2: Performance (expected discounted reward) of IQL and Q-learning in IPD for a range of exploration rates.

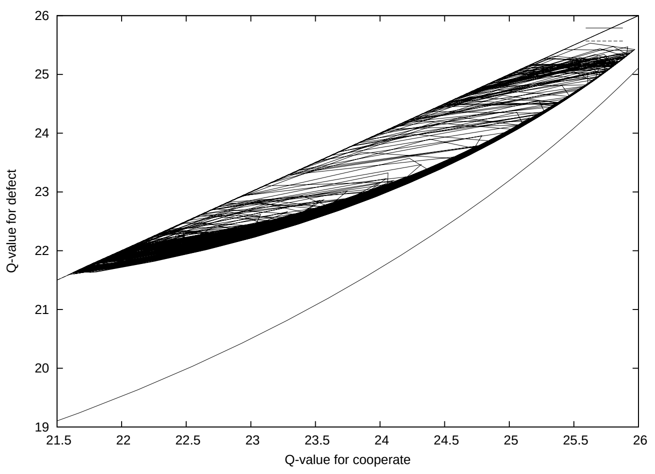

Figure 3: A trajectory of the Q values for cooperation vs. defection for an independent IQL agent. Instead of converging, the values enter a narrow region and then rattle around there.

orbit—a loop traversed repeatedly. In the latter case, the average reward along the loop exactly matches that of a Nash equilibrium. Although IGA is not directly realizable because of the need for arbitrarily small learning rates, it did give rise to several practical algorithms such as WoLF-IGA, GIGA [22], and GIGA-WoLF [5].

In this section, we view Q-learning in a similar way. We define Infinitesimal Q-learning (IQL), a version of Q-learning where value updates are arbitrarily small. Whereas Q-learning (given any amount of exploration) can only converge to mutual best responses (Nash equilibria), we find that IQL includes non-Nash fixed points and that the location of these fixed points is similar to the values obtained by standard Q-learning.

IQL for two-player, two-action games is defined as follows. Let a^* be the action that maximizes the Q value for player i, $\bar{a}$ be the other action, b^* be the action that maximizes the Q value for the opponent, and $\bar{b}$ be the opponent's other action. Then,

$$\begin{array}{llll} Q_i^{a^*} \leftarrow Q_i^{a^*} & +\alpha(1-\frac{\epsilon}{2})^2 & (R_i^{a^*b^*} + \gamma Q_i^{a^*} - Q_i^{a^*}) \\ & +\alpha(1-\frac{\epsilon}{2})\frac{\epsilon}{2} & (R_i^{a^*\bar{b}} + \gamma Q_i^{a^*} - Q_i^{a^*}) \\ Q_i^{\bar{a}} \leftarrow Q_i^{\bar{a}} & +\alpha\frac{\epsilon}{2}(1-\frac{\epsilon}{2}) & (R_i^{\bar{a}b^*} + \gamma Q_i^{a^*} - Q_i^{\bar{a}}) \\ & +\alpha\frac{\epsilon^2}{4} & (R_i^{\bar{a}\bar{b}} + \gamma Q_i^{a^*} - Q_i^{\bar{a}}). \end{array}$$

The idea here is that the Q values, for sufficiently small values of the learning rate α, explore in all directions simultaneously, with the resulting update weighted by its probability. For example, with the maximum probability $(1-\epsilon)^2$, both players will choose their greedy actions and update their values in the direction of the resulting payoffs. But, with smaller probability, one or the other agent will explore, resulting in a different update. The IQL update rule blends these updates together based on how often they occur given the exploration-rate parameter ϵ.

Note IQL is a deterministic algorithm—two IQL agents, starting with identical Q functions, will remain sychronized as they make the same series of updates.

Figure 2 shows the result of IQL in IPD with a range of exploration-rate parameters (discount $\gamma = 0.9$). In each case, starting roughly from mutual cooperation Q values, IQL converges. As long as exploration stays low, the temptation of defection contributes very little to the update and the higher values of cooperation keep the Q values high. Once the exploration rate is high enough, though, the values for defection are updated frequently, overtake the values for cooperation, and mutual defection becomes the only stable solution.

Note that, for the low exploration rates, the converged value of cooperation and defection are equal. This result is not possible in Q-learning, since the value for defection is always one larger than cooperation. So, against any fixed opponent, Q-learning will learn distinct values for these two actions. However, in IQL, the rare defection explorations are balanced by the more common mutual cooperation scores.

For comparison, we carried out a similar experiment with Q-learning. For each exploration rate from 0.02 to 1 (stepping by .02), we intiailized two Q-learners with Q values of 40 for both actions. We ran them with a low fixed learning rate of $\alpha = .0001$ for 1,000,000 steps and recorded the Q values for cooperation and defection from the first agent. Note the remarkable similarity between the outcomes of Q-learning and IQL in this case.

An important difference between IQL and Q-learning is that the randomness in exploration in Q-learning makes it impossible for two learners to stay completely synchronized. We have found that two IQL agents tend not to produce convergent values, but instead oscillate chaotically. Figure 3 shows the trajectory of values reached by IQL in IPD with an exploration rate of $\epsilon = 0.2$. Figure 4 provides another view of the data, showing the Q value for the cooperate action for the two players over time. Notice how one player consistently has slightly higher values, which may be part of the reason the values don't converge.

4. DIRECT ANALYSIS OF IPD

In this section, we present a different analysis of the behavior of Q-learning in the IPD from the perspective of the probabilities of various events that change the Q values for the two players.

The analysis makes use of the following quantities:

- Q_i^C : Q value of C for player i
- Q_i^D : Q value of D for player i

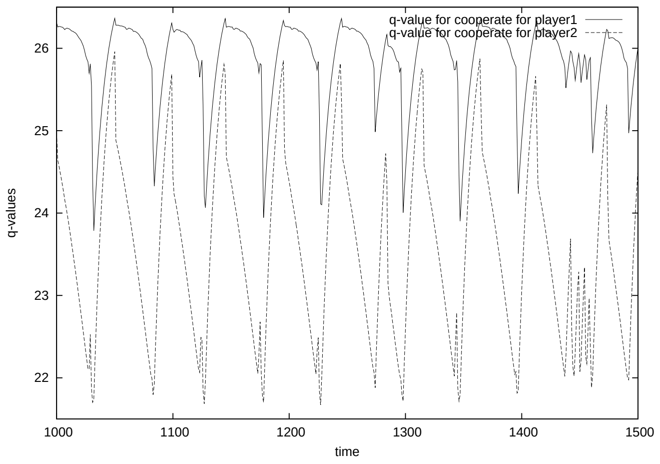

Figure 4: The cooperate Q value for a pair of simultaneously IQL agents over time, showing the chaotic pattern of rises and falls.

- $Q^C(DD)$: converged Q value at DD equilibrium
- ϕ_{C_j}: fraction that C is chosen by player j, above exploration. How likely is it that the greedy action is C?
- $\phi_{D_j} = 1 - \phi_{C_j}$
- $Q_i^C(\phi_{C_j})$: converged Q value of player i when player j plays C at fraction ϕ_{C_j}.

We can express the converged Q values as:

$$\begin{aligned}
Q_C^{DD*} &= \frac{S(1-\epsilon)+R\epsilon}{1-\gamma} \\
Q_D^{DD*} &= \frac{P(1-\epsilon)+T\epsilon}{1-\gamma} \\
Q_C^{CC*} &= \frac{R(1-\epsilon)+S\epsilon}{1-\gamma} \\
Q_D^{CC*} &= \frac{T(1-\epsilon)+P\epsilon}{1-\gamma} \\
Q_C^{\phi_{C_j}} &= \frac{(R\phi_{C_j}+S\phi_{D_j})(1-\epsilon)+(R\phi_{D_j}+S\phi_{C_j})\epsilon}{1-\gamma} \\
Q_D^{\phi_{C_j}} &= \frac{(T\phi_{C_j}+P\phi_{D_j})(1-\epsilon)+(T\phi_{D_j}+P\phi_{C_j})\epsilon}{1-\gamma}.
\end{aligned}$$

Switching between mutual values is dictated by:

$$\begin{aligned}
\Pr(DD \to CC) &= O(\min(\epsilon^2, \epsilon^{2\frac{Q_D^{DD*}-Q_C^{DD*}}{R\alpha}})) \\
\frac{\partial Q}{\partial \alpha}\frac{\partial \alpha}{\partial t} &= \frac{\partial Q}{\partial t} \\
\delta_C = \frac{\partial Q_{Ci}}{\partial t} &= (\gamma-1)Q_{Ci} + R(\phi_{C_j}(1-\epsilon)+\phi_{D_j}\epsilon) + \\
& \quad S(\phi_{D_j}(1-\epsilon)+\phi_{C_j}\epsilon) \\
\delta_D = \frac{\partial Q_{Di}}{\partial t} &= (\gamma-1)Q_{Di} + T(\phi_{C_j}(1-\epsilon)+\phi_{D_j}\epsilon) + \\
& \quad P(\phi_{D_j}(1-\epsilon)+\phi_{C_j}\epsilon).
\end{aligned}$$

If the other player only plays C and the Q values are at the same level, when is $\delta_C(1-\epsilon) \geq \delta_D\epsilon$?

The answer should be given when the ratio of the distances the algorithms are attempting to close between the two Q values is equal to the exploration rate. The rate at which either Q value is changed is proportional to the distance to the goal (given some action by the other player). The rate at which the under-value is chosen is equal to the exploration rate. Therefore:

$$(1-\epsilon)(\frac{R(1-\epsilon)+S\epsilon}{1-\gamma} - Q_C) \geq \epsilon(\frac{T(1-\epsilon)+P\epsilon}{1-\gamma} - Q_D).$$

We can put additional constraints on the values:

$$\begin{aligned}
Q_C &= Q_D \\
(1-2\epsilon)Q_C &\leq (1-\epsilon)(\frac{R(1-\epsilon)+S\epsilon}{1-\gamma}) - \epsilon(\frac{T(1-\epsilon)+P\epsilon}{1-\gamma}) \\
(1-2\epsilon)Q_C &\leq \frac{-T\epsilon(1-\epsilon)+R(1-\epsilon)^2-P\epsilon^2+S\epsilon(1-\epsilon)}{1-\gamma} \\
Q_C &\leq \frac{-T\epsilon(1-\epsilon)+R(1-\epsilon)^2-P\epsilon^2+S\epsilon(1-\epsilon)}{(1-\gamma)(1-2\epsilon)} \\
Q_C &\leq \frac{-4(0.1)(0.9)+3(0.9)^2-0(0.1)^2-1(0.1)(0.1)}{(0.1)(0.8)} \\
Q_C &\leq 25.75.
\end{aligned}$$

Above this value, D has the upper hand and it will assume control as soon as it gets the chance. However, this analysis assumes that the other player is in a constant C mode, which is not quite right. This value simply represents the ultimate ceiling for the values we have chosen, and where the learning rate is very small. When the other player's action mix is more varied, the place where both rates are equal shifts. When the opponent's mix proportion is not constant either, then you have the dynamics we have seen.

Here are the real deltas:

$$\begin{aligned}
\delta_C &= (\gamma-1)Q_{Ci} + R(\phi_{C_j}(1-\epsilon)+\phi_{D_j}\epsilon) + \\
& \quad S(\phi_{D_j}(1-\epsilon)+\phi_{C_j}\epsilon) \\
\delta_D &= (\gamma-1)Q_{Di} + T(\phi_{C_j}(1-\epsilon)+\phi_{D_j}\epsilon) + \\
& \quad P(\phi_{D_j}(1-\epsilon)+\phi_{C_j}\epsilon).
\end{aligned}$$

To find out what the true fractions/target values are currently, we need to start somewhere. Here is one way to do it. We are going on the assumption that $Q_C = Q_D$ as we have seen.

At what fraction of C ϕ_{C_j} from the other player is the target already met at the current value of Q_C? Where $\delta_C = 0$:

$$\begin{aligned}
(1-\gamma)Q_{Ci} &= \phi_{C_j}R(1-\epsilon) + (1-\phi_{C_j})R\epsilon + \\
& \quad (1-\phi_{C_j})S(1-\epsilon) + \phi_{C_j}S\epsilon \\
(1-\gamma)Q_{Ci} &= \phi_{C_j}R(1-2\epsilon) + R\epsilon + \\
& \quad (1-\phi_{C_j})S + S(1-\epsilon) \\
\phi_{C_j}(R(1-2\epsilon)-S) &= (1-\gamma)Q_{Ci} - R\epsilon - S(2-\epsilon) \\
\phi_{C_j} &= \frac{(1-\gamma)Q_{Ci} - R\epsilon - S(2-\epsilon)}{R(1-2\epsilon)-S}.
\end{aligned}$$

If we have $Q_C = 25$, for example, with R = 3 and S = 0:

$$\begin{aligned}\phi_{C_j} &= \frac{(.1)25 - 3(0.1)}{3(0.8)}\\ \phi_{C_j} &= \frac{2.2}{2.4} = 11/12.\end{aligned}$$

This value is not the end of the story. We find both fractions, then compute the new targets given by those fractions, which then lead to new fractions, and so on. This method is one way to discover the point at which the process will start repeating again. Only then will we know the fraction resulting at a level of Q value. In any event, it is clear that the Q-learning update rule can lead to non-equilibrium values, specifically when the off-equilibrium behavior attains higher reward.

5. FUTURE WORK

This work is still in progress. We would like to provide a precise characterization of the behavior of IQL. Does it converge for some initial Q values? Or is it always chaotic? Can we characterize the range of values encountered during the chaotic oscillations? We are interested in applying tools from non-linear dynamics to try to modify IQL to make it behave more consistently. If an appropriate tool is found, we plan to insert it into Q-learning to modify the exploration or learning rate to produce a more robust algorithm. It is apparent that the seeds of a powerful approach already exist in a simple form in Q-learning and we would like to provide a more reliable alternative.

6. REFERENCES

[1] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

[2] M. Babes, E. Munoz de Cote, and M. L. Littman. Social reward shaping in the prisoner's dilemma. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1389–1392, 2008.

[3] A. Bonarini, A. Lazaric, J. E. Muñoz de Cote, and M. Restelli. Improving cooperation among self-interested reinforcement learning agents. In *Proceedings of ECML'05 Workshop on Reinforcement Learning in Non-Stationary Environments*, pages 25–36, 2005.

[4] M. Bowling and M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

[5] M. H. Bowling. Convergence and no-regret in multiagent learning. In *NIPS*, 2004.

[6] A. Greenwald and K. Hall. Correlated-Q learning. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 242–249, 2003.

[7] J. Hu and M. P. Wellman. Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003.

[8] T. Jaakkola, M. I. Jordan, and S. P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 6(6):1185–1201, 1994.

[9] M. L. Littman. Friend-or-foe Q-learning in general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 322–328. Morgan Kaufmann, 2001.

[10] M. L. Littman and P. Stone. Implicit negotiation in repeated games. In *Eighth International Workshop on Agent Theories, Architectures, and Languages (ATAL-2001)*, pages 393–404, 2001.

[11] M. L. Littman and C. Szepesvári. A generalized reinforcement-learning model: Convergence and applications. In L. Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 310–318, 1996.

[12] M. Mundhe and S. Sen. Evaluating concurrent reinforcement learners. In *Proceedings of the Fourth International Conference on Multiagent Systems*, pages 421–422. IEEE Press, 2000.

[13] E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 880–887, 2004.

[14] T. W. Sandholm and R. H. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems*, 37:144–166, 1995.

[15] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 39:287–308, 2000.

[16] S. Singh, M. Kearns, and Y. Mansour. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of Uncertainty in AI (UAI)*, 2000.

[17] A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman. PAC model-free reinforcement learning. In *Proceedings of the Twenty-third International Conference on Machine Learning (ICML-06)*, 2006.

[18] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[19] J. N. Tsitsiklis. Asynchronous stochastic approximation and Q-learning. *Machine Learning*, 16(3):185–202, 1994.

[20] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, 1989.

[21] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[22] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.

[23] M. Zinkevich, A. R. Greenwald, and M. L. Littman. Cyclic equilibria in Markov games. In *Advances in Neural Information Processing Systems 18*, 2005.