

Reinforcement Learning for Average Reward Zero-Sum Games

Shie Mannor

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology, Cambridge, MA 02139
shie@mit.edu

Abstract. We consider Reinforcement Learning for average reward zero-sum stochastic games. We present and analyze two algorithms. The first is based on relative Q-learning and the second on Q-learning for stochastic shortest path games. Convergence is proved using the ODE (Ordinary Differential Equation) method. We further discuss the case where not all the actions are played by the opponent with comparable frequencies and present an algorithm that converges to the optimal Q-function, given the observed play of the opponent.

1 Introduction

Since published in [DW92], the Q-learning algorithm was implemented in many applications and was analyzed in several different setups (e.g., [BT95,ABB01,BM00]). The Q-learning algorithm for learning an optimal policy in Markov Decision Processes (MDPs) is a direct off-policy learning algorithm in which a Q-value vector is learned for every state and action. For the discounted case, the Q-value of a specific state-action pair represents the expected discounted utility if the action is chosen in the specific state and an optimal policy is then followed. In this work we deviate from the standard Q-learning scheme in two ways. First, we discuss games, rather than MDPs. Second, we consider the average reward criterion rather than discounted reward.

Reinforcement learning for average reward MDPs was suggested in [Sch93] and further studied in [Sin94,Mah96]. Some analysis appeared later in [ABB01,BT95]. The analysis for average reward is considerably more cumbersome than that of discounted reward, since the dynamic programming operator is no longer a contraction. There are several methods for average reward reinforcement learning, including Q-learning ([ABB01]), a polynomial PAC model-based learning model ([KS98]), actor critic ([KT03]), etc. Convergence proofs of the Q-learning algorithms for average reward typically rely on the ODE method and the fact that the Q-learning algorithm is essentially an asynchronous stochastic approximation algorithm.

Q-learning for zero-sum stochastic games (SGs) was suggested in [Lit94] for discounted reward. The convergence proof of this algorithm appears, in a broader context, in [LS99]. The main difficulty in applying Q-learning to games is that

Q-learning is inherently an off-policy learning algorithm. This means that the optimal policy is learned while another policy is played. Moreover, the opponent may refrain from playing certain actions (or play them only a few times) so the model parameters may never be fully revealed. Consequently, every learning algorithm is doomed to learn a potentially inferior policy. On-policy algorithms, whose performance is measured according to the reward they accumulate may, however, attain an average reward which is close to the value of the game (e.g., [BT02]). We note two major difficulties with Q-learning style algorithms. First, one needs all actions in all states to be chosen infinitely often by both players (actually comparatively often for average reward). Second, the standard analysis of Q-learning (e.g., [Tsi94,BT95]) relies on contraction properties of the dynamic programming operator which follow easily for discounted reward or shortest path problems, but do not hold for average reward. We start by addressing the second issue and present two Q-learning type algorithms for SGs. We show that if all actions in all states are played comparatively often then convergence to the true Q-value is guaranteed. We then tackle the problem of exploration and show that by slightly modifying the Q-learning algorithm we can make sure that the Q-vector converges to the Q-vector of the observed game.

The convergence analysis of the Q-learning algorithms is based on [BM00,ABB02]. The main problem is the unfortunate fact that the dynamic programming operator of interest is not a contraction operator. In Section 3 we present a version of Relative Q-learning (e.g., [BS98]) adapted to average reward SGs. We later modify the λ -SSP (Stochastic Shortest Path) formulation of [BT95, Section 7.1] to average reward SGs. The idea is to define a related SSPG (Stochastic Shortest Path Game) and show that by solving the SSPG the original average reward problem is solved as well.

The paper is organized as follows: In Section 2 we define the SG model, and recall some results from the theory of stochastic games. The relative Q-learning algorithm for average reward games is presented in Section 3. The λ -SSPG algorithm is described in Section 4. Since the opponent may refrain from playing certain actions, the true Q vector may be impossible to learn. We show how this can be corrected by concerning the observed game. This is done in Section 5. Brief concluding remarks are drawn in Section 6. The convergence proofs of both algorithms are deferred to the appendix.

2 Model and Preliminaries

In this section we formally define SGs. We then state a stability assumption which is needed in order to guarantee that our analysis holds and that the value is independent of the initial state. We finally survey some known results from the theory of SGs.

2.1 Model

We consider an average reward zero-sum finite (states and action) Stochastic Game (SG) which is played ad-infinitum. We refer to the players as P1 (the

decision maker in interest) and P2 (the adversary). The game is defined by the five-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{B}, P, r)$, where:

1. \mathcal{S} is the finite set of states of the stochastic game, $\mathcal{S} = \{1, \dots, S\}$.
2. \mathcal{A} is the set of actions of P1 in each state, $\mathcal{A} = \{1, \dots, A\}$. To streamline the notations it is assumed that in all states P1 has the same available actions.
3. \mathcal{B} is the set of actions of P2 in each state, $\mathcal{B} = \{1, \dots, B\}$. It is assumed that in all states P2 has the same available actions.
4. P is the conditional transition law. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \times \mathcal{S} \rightarrow [0, 1]$ such that $P(s'|s, a, b)$ is the probability that the next state is s' given that current state is s , P1 plays a and P2 plays b .
5. r is P1's (random) reward function, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{B} \mapsto \mathbb{R}$. The reward obtained when P1 plays a , P2 plays b and the current state is s is distributed according to a measure $\mu(s, a, b)$ whose mean is $r(s, a, b)$. A bounded second moment is assumed.

At each time epoch n , both players observe the current state s_n , and then P1 and P2 choose actions a_n and b_n , respectively. As a result P1 receives a reward of r_n which is distributed according to $\mu(s_n, a_n, b_n)$. The next state is determined according to the transition probability $P(\cdot|s_n, a_n, b_n)$. A policy $\sigma_1 \in \Sigma_1$ for P1 is a mapping from all possible histories (including states, actions, and rewards) to the set of mixed actions $\Delta(\mathcal{A})$, where $\Delta(\mathcal{A})$ is the set of all probability measures over \mathcal{A} . Similarly, a policy $\sigma_2 \in \Sigma_2$ for P2 is a mapping from all possible histories to the mixed actions $\Delta(\mathcal{B})$. A policy of either player is called *stationary* if the mixed action in time n depends only on the state s_n . Let the average reward at time n be denoted by $\hat{r}_n \triangleq \sum_{\tau=1}^n r_\tau/n$.

2.2 A Stability Assumption

We shall make the following assumption throughout the paper. The assumption can be thought of as a stability or recurrence assumption. The state s^* is a reference state to which a return is guaranteed. Recall that a state is recurrent under a certain pair of policies of P1 and P2 if that state is visited with probability 1 in finite time when the players follow these policies.

Assumption 1 (Recurrent State) *There exists a state $s^* \in \mathcal{S}$ which is recurrent for every pair of stationary strategies played by P1 and P2.*

We say that an SG has a value v if

$$v = \sup_{\sigma_1} \inf_{\sigma_2} \liminf_{n \rightarrow \infty} \mathbb{E}_{\sigma_1, \sigma_2}[\hat{r}_n] = \inf_{\sigma_2} \sup_{\sigma_1} \limsup_{n \rightarrow \infty} \mathbb{E}_{\sigma_1, \sigma_2}[\hat{r}_n].$$

For finite games, the value exists ([MN81]). If Assumption 1 holds, then the value is independent of the initial state and can be achieved in stationary strategies (e.g., [FV96]).

2.3 Average Reward Zero-Sum Stochastic Games - Background

We now recall some known results from the theory of average reward scalar games. We assume henceforth that Assumption 1 is satisfied. For such games it is known (e.g., [FV96, Theorem 5.3.3]) that there is a value and a bias vector, that is there exists a number v and a vector $H \in \mathbb{R}^S$ such that for each $s \in S$:

$$H(s) + v = \operatorname{val}_{a,b} \left[r(s, a, b) + \sum_{s' \in S} p(s'|s, a, b)H(s') \right], \quad (2.1)$$

where $\operatorname{val}_{a,b}$ is the minimax operator, that is for a matrix R with A rows and B columns the $\operatorname{val}_{a,b}$ operator is: $\operatorname{val}_{a,b}[R] \triangleq \inf_{v \in \Delta(A)} \sup_{u \in \Delta(B)} \sum_{a=1}^A \sum_{b=1}^B v_a u_b R_{ab}$. Furthermore, in [Pat97, page 90] it was shown that under Assumption 1 there exists a unique H such that Equation (2.1) holds for every $s \in S$ and for some specific s' we have that $H(s') = 0$. We note that when the game parameters are known there are efficient methods to find the H and v ; see [FV96, Pat97]. It is often convenient to use operator notations. In this case the resulting (vector) equation is:

$$ve + H^* = TH^*, \quad (2.2)$$

where $e \in \mathbb{R}^S$ is the ones vector ($e = (1, \dots, 1)$) and $T : \mathbb{R}^S \mapsto \mathbb{R}^S$ is the dynamic programming operator defined by:

$$TH(s) \triangleq \operatorname{val}_{a,b} \left[\sum_{s' \in S} r(s, a, b) + P(s'|s, a, b)H(s') \right]. \quad (2.3)$$

It turns out that T is not a contraction, so that Q-learning style mechanisms that rely on contraction properties may not converge. Thus, a refined scheme should be developed. Note that if H^* is a solution of (2.2) so is $H^* + ce$, so that one must take into account the non uniqueness of the solutions of (2.2). We propose two different schemes to overcome this non-uniqueness. The first scheme is based on the uniqueness of the solution of Equation (2.2) that satisfies $H(s^*) = 0$, and the second is based on a contraction property of a related dynamic programming operator (for an associated stochastic shortest path game).

Our goal is to find the optimal Q-vector which satisfies that: $Q^*(s, a, b) \triangleq r(s, a, b) + \sum_{s'} p(s'|s, a, b)H^*(s')$, where H^* is a solution of the optimality equation (2.2). Note that if H^* is determined uniquely (by requiring $H^*(s^*) = 0$) then Q^* is also unique. The Q-vector is defined on $\mathbb{R}^{S \cdot A \cdot B}$, the interpretation of $Q(s, a, b)$ is the relative gain for P1 to use action a assuming P2 will use action b , when current state is s . Given the vector Q^* , the maximin policy is to play at state s a maximin (mixed) action with respect to the matrix game whose entries are $Q(s, \cdot, \cdot)$.

3 Relative Q-learning

Relative Q-learning for average reward MDPs was suggested by [Sch93], and studied later in [Sin94, Mah96]. It is the simulation counterpart of the relative

value iteration algorithm (e.g., [Put94]) for solving average reward MDPs. The following algorithm is the SG (asynchronous) version of the relative Q-learning algorithm.

$$Q_{n+1}(s, a, b) = Q_n(s, a, b) + \mathbf{1}_{\{s_n=s, a_n=a, b_n=b\}} \gamma (N(n, s, a, b)) \left(r_n + FQ_n(s_{n+1}) - f(Q_n) - Q_n(s, a, b) \right), \quad (3.4)$$

where $N(n, s, a, b)$ denote the number of times that state s and actions a and b were played up to time n (i.e., $N(n, s, a, b) = \sum_{\tau=1}^n \mathbf{1}_{\{s_\tau=s, a_\tau=a, b_\tau=b\}}$), and $F : \mathbb{R}^{S \cdot A \cdot B} \mapsto \mathbb{R}^S$ is the per state value function which satisfies: $FQ(s) \triangleq \text{val}_{a,b}[Q(s, a, b)]$. The function $f(Q) : \mathbb{R}^{S \cdot A \cdot B} \rightarrow \mathbb{R}$ is required to have the following properties: 1. f is Lipschitz; 2. f is scaling invariant — $f(aQ) = af(Q)$; 3. f is translation invariant — $f(Q + er) = f(Q) + r$ where e is the vector of ones (note the abuse of notations - e is $RSAB$ dimensional here). Examples for valid f 's are $f(Q) = Q(s^0, a^0, b^0)$ for some (s^0, a^0, b^0) or $f(Q) = \frac{1}{SAB} \sum_{s,a,b} Q(s, a, b)$. Intuitively, f takes care of having the Q-vector bounded. More precisely, we shall use f in the proof to ensure that the underlying ODE has a unique solution.

We require the standard stochastic approximation assumption on the learning rate γ . Namely, γ should be square summable but not summable, and “regular” in the sense that it does not vanish occasionally. More precisely:

Assumption 2 (Learning Rate) *The sequence $\gamma(n)$ satisfies:¹*

1. For every $0 < x < 1$, $\sup_k \gamma(\lfloor xk \rfloor) / \gamma(k) < \infty$.
2. $\sum_{n=1}^{\infty} \gamma(n) = \infty$ and $\sum_{n=1}^{\infty} \gamma(n)^2 < \infty$.
3. For every $0 < x < 1$ the limit $(\sum_{m=1}^{\lfloor yn \rfloor} \gamma(m)) / (\sum_{m=1}^n \gamma(m)) \rightarrow 1$ uniformly in $y \in [x, 1]$.

For example, $\gamma(n) = 1/n$ and $1/n \log n$ ($n > 1$) satisfy this assumption. The following assumption is crucial in analyzing the asynchronous stochastic approximation algorithm.

Assumption 3 (Often updates) *There exists a deterministic $\delta > 0$ such that for every $s \in \mathcal{S}, a \in \mathcal{A}, b \in \mathcal{B}$, $\liminf_{n \rightarrow \infty} \frac{N(n, s, a, b)}{n} \geq \delta$ with probability 1. That is, all component are updated comparatively often.*

The following theorem is proved in Appendix A.1.

Theorem 1. *Suppose that Assumptions 1, 2 and 3 hold. Then the asynchronous algorithm (3.4) converges with probability 1 to Q^* .*

4 λ -SSPG Q-learning

A different approach is to use the λ -SSP (Stochastic Shortest Path) formulation, suggested by Bertsekas and Tsitsiklis [BT95, Section 7.1] for average reward

¹ $\lfloor x \rfloor$ is the integer part of x .

MDPs and analyzed in [ABB01]. The key idea is to view the average reward as the ratio of the expected total reward between renewals and the expected time between renewals. We consider a similar approach for SGs, using results from [Pat97] regarding SSPGs. From the stochastic approximation point of view we maintain two time scales. We iterate the average reward estimate, λ , slowly towards the value of the game, while the Q-vector is iterated on a faster scale so that it tracks the Q-vector of the associated SSPG. The convergence follows from Borkar's two-time-scale stochastic approximation ([Bor97]). There are two equations that are iterated simultaneously, the first is related to the Q-vector, is defined as a vector in $\mathbb{R}^{S \cdot A \cdot B}$ and the second is related to λ which is a real number. The λ -SSPG Q-learning algorithm is:

$$\begin{aligned} Q_{n+1}(s, a, b) &= Q_n(s, a, b) + \gamma(N(n, s, a, b)) \left(r_n + FQ_n(s_{n+1}) 1_{\{s_{n+1} \neq s^*\}} \right. \\ &\quad \left. - \lambda_n - Q_n(s, a, b) \right) 1_{\{s_n = s, a_n = a, b_n = b\}} \\ \lambda_{n+1} &= \Lambda[\lambda_n + b(n)FQ_n(s^*)], \end{aligned} \tag{4.5}$$

where $b(n) = o(\gamma(n))$, Λ is the projection to the interval $[-K, K]$ chosen such that $|\nu| < K$, and $N(n, s, a, b)$ and F are as before.

An additional assumption we require is that all the elements are sampled in an evenly distributed manner. More precisely:

Assumption 4 For every $x > 0$ let $M(n, x) = \min\{m \geq n : \sum_{k=n}^m \gamma(k) \geq x\}$, for every $s, s' \in \mathcal{S}$, $a, a' \in \mathcal{A}$, $b, b' \in \mathcal{B}$ the limit:

$$\lim_{n \rightarrow \infty} \frac{\sum_{k=N(n, s, a, b)}^{N(n, x, s, a, b)} \gamma(k)}{\sum_{k=N(n, s', a', b')}^{N(n, x, s', a', b')} \gamma(k)}$$

exists almost surely.

The following theorem is proved in Appendix A.2.

Theorem 2. Suppose that Assumptions 1, 2, 3, and 4 hold. Further, assume that $b(n)$ satisfies Assumption 2 and that $b(n) = o(\gamma(n))$. Then the asynchronous algorithm (4.5) converges with probability 1 so that $Q_n \rightarrow Q^*$ and $\lambda_n \rightarrow \nu$.

5 The Often Update Requirement

The convergence of both algorithms described in the previous sections required several assumptions. Assumption 1 is a property of the (unknown) game. Assumption 2 is controlled by P1's choice of the learning rate and can be easily satisfied. Assumption 3 (and 4 for the second algorithm) presents an additional difficulty. The often updates requirement restricts not only on P1's policy but also P2's actual play. Obviously, P1 cannot impose on P2 to perform certain actions and consequently we cannot guarantee that $Q_n \rightarrow Q^*$. In this section we

consider methods to relax the often updates assumption. We will suggest a modification of the relative Q-learning algorithm to accommodate for state-action-action triplets that are not played comparatively often. We start with discussing possible approaches to handle the case where not all state-action-action triplets are sampled comparatively often.

If certain state-action-action triplets are performed finitely often their Q-values cannot be learned (since even the estimation of the immediate reward is not consistent). We therefore must restrict the attention of the learning algorithm to Q-value of triplets that are played infinitely often, and make sure that the triplets that are not played often do not interfere with the estimation of the Q-value of the other triplets. The main problem is that we do not know (at any given time) if an action will be chosen finitely often (and can be ignored) or comparatively often (and should be used in the Q-learning). We therefore suggest to maintain a set of triplets that have been played often enough, and essentially learn only on this set. Let $Y_n(\delta)$ denote the set of triplets that were sampled more than δ fraction of the time up to time n , that is: $Y_n(\delta) \triangleq \{(s, a, b) \in \mathcal{S} \times \mathcal{A} \times \mathcal{B} : \frac{N(n, s, a, b)}{n} \geq \delta\}$. The algorithm we suggest is the following modification of (3.4):

$$Q_{n+1}(s, a, b) = \begin{cases} Q_n(s, a, b) + 1_{\{s_n=s, a_n=a, b_n=b\}} \gamma(N(n, s, a, b)) (r_n + FQ_n(s_{n+1}) - f(Q_n) - Q_n(s, a, b)) & \text{if } (s, a, b) \in Y_n(\delta) \\ -M & \text{if } (s, a, b) \notin Y_n(\delta) \end{cases} \quad (5.6)$$

where M is a large positive number which is larger than $\max_{s,a,b} |Q(s, a, b)|$. Let

$$Y_\infty(\delta) = \{(s, a, b) \in \mathcal{S} \times \mathcal{A} \times \mathcal{B} : \liminf_n \frac{N(n, s, a, b)}{n} \geq \delta\}$$

denote the set of triplets that are chosen comparatively often (δ is a deterministic constant). We refer to the game which is restricted to triplets in $Y_\infty(\delta)$ as the δ -observed game. We denote the solution of Bellman's equation (2.3) where the a, b entry for all the triplets not in $Y_\infty(\delta)$ is replaced by $-M$ (and are therefore not relevant to the optimal policy) by $H_{Y_\infty}^*$ and the matching Q-vector by $Q_{Y_\infty}^*$.

Theorem 3. *Suppose that Assumptions 1 and 2 hold, and suppose that for every state-action-action triplet (s, a, b) we have that:*

$$\liminf_{n \rightarrow \infty} \frac{N(n, s, a, b)}{n} \geq \delta \quad \text{or} \quad \limsup_n \frac{N(n, s, a, b)}{n} < \delta.$$

Then (5.6) converges with probability one so that $Q_n(s, a, b) \rightarrow Q_{Y_\infty}^$ for every $(s, a, b) \in Y_\infty(\delta)$.*

Proof. For every triplet (s, a, b) in $Y_\infty(\delta)$ there exists a time $\tau(s, a, b)$ such that for every $n > \tau(s, a, b)$ the triplet $(s, a, b) \in Y_n(\delta)$. By the condition in the theorem if $(s, a, b) \notin Y_\infty(\delta)$ then there exists a time $\tau'(s, a, b)$ such that for every $n >$

$\tau'(s, a, b)$ the triplet $(s, a, b) \notin Y_n(\delta)$. Let $\tau = \max\{\max_{(s,a,b) \in Y_\infty(\delta)} \tau(s, a, b), \max_{(s,a,b) \notin Y_\infty(\delta)} \tau'(s, a, b)\}$. Suppose now that the learning algorithm begins at time τ . Since τ is finite it is easy to see that Assumptions 1-3 are satisfied restricted to $(s, a, b) \in Y_\infty(\delta)$ so that by Theorem 1 the result follows. Note that the triplets which are not in $Y_\infty(\delta)$ are updated every epoch (after τ) with the value $-M$. \square

Naturally, some actions may satisfy neither the lim inf nor the lim sup conditions. A method that controls δ dynamically, and allows to circumvent this problem is under current study.

6 Concluding Remarks

We presented two Q-learning style algorithms for average reward zero-sum SGs. Under appropriate recurrence and often updates assumptions the convergence to the optimal policy was established. Our results generalize the discounted case that was proved in [LS99]. There are several open questions that warrants further study. First, the extension of the results presented in this paper to games with a large state space where function approximation is needed appears non-trivial. Second, we only partially addressed the issue of actions that are not chosen comparatively often by the Q-learning algorithm. There are several other possibilities that can be considered (using a promotion function as in [EDM01], adding bias factor as in [LR85], and optimistic initial conditions as in [BT02]) none have proved a panacea for the complications introduced by “uneven” exploration. Third, we only considered zero-sum games. Extending the algorithms presented here to general sum games appears difficult (even the extension for discounted reward is a daunting task). Finally, universal consistency in SGs (e.g., [MS03]) is a related challenging problem. In this setup P1 tries to attain an average reward which is as high as the average reward that could have been attained had P2’s strategy (or some statistical measure thereof) was provided in advance. The definitions for universal consistency in SGs are involved and the strategies suggested to date are highly complex. Devising a simple algorithm in the style of Q-learning is of great interest. We note, however, that the distinctive property of universal consistency is that P2 strategy cannot be assumed stationary, so stochastic approximation algorithms which rely on stationarity may not work.

References

- [ABB01] J. Abounadi, D. Bertsekas, and V. Borkar. Learning algorithms for Markov decision processes with average cost. *SIAM J. Control Optim.*, 40:681 – 698, 2001.
- [ABB02] J. Abounadi, D. Bertsekas, and V. Borkar. Stochastic approximation for non-expansive maps: Application to Q-learning algorithms. *SIAM J. Control Optim.*, 41:1–22, 2002.
- [BM00] V.S. Borkar and S.P. Meyn. The O.D.E. method for convergence of stochastic approximation and reinforcement learning. *SIAM J. Control Optim.*, 38(2):447–469, 2000.

- [Bor97] V.S. Borkar. Stochastic approximation with two time scales. *IEEE systems and Control letters*, 29:291–294, 1997.
- [Bor98] V.S. Borkar. Asynchronous stochastic approximation. *SIAM J. Control Optim.*, 36:840–851, 1998.
- [BS98] A.G. Barto and R.S. Sutton. *Reinforcement Learning*. MIT Press, 1998.
- [BS99] V.S. Borkar and K. Soumyanath. An analog scheme for fixed point computation - part I: Theory. *IEEE Trans. On Circuits and Systems*, 44(4):7–13, April 1999.
- [BT95] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1995.
- [BT02] R.I. Brafman and M. Tennenholtz. R-MAX, a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2002.
- [DW92] P. Dayan and C. Watkins. Q-learning. *Machine Learning*, 8:279–292, 1992.
- [EDM01] E. Even-Dar and Y. Mansour. Convergence of optimistic and incremental Q-learning. In *NIPS*, 2001.
- [FV96] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer Verlag, 1996.
- [KB00] V. Konda and V. Borkar. Actor-critic-type algorithms for Markov decision problems. *SIAM J. Control Optim.*, 38:94–123, 2000.
- [KS98] M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. In *Proceedings of the 15th International Conference on Machine Learning*, pages 260–268. Morgan Kaufmann, 1998.
- [KT03] V. R. Konda and J. N. Tsitsiklis. Actor-critic algorithms. *SIAM J. Control Optim.*, 42(4):1143–1166, 2003.
- [KY97] H.J. Kushner and C.J. Yin. *Stochastic Approximation Algorithms and Applications*. Springer Verlag, 1997.
- [Lit94] M.L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 157–163. Morgan Kaufman, 1994.
- [LR85] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.
- [LS99] M.L. Littman and C. Szepesvári. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Computation*, 11(8):2017–2059, 1999.
- [Mah96] S. Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22(1):159–196, 1996.
- [MN81] J.F. Mertens and A. Neyman. Stochastic games. *International Journal of Game Theory*, 10(2):53–66, 1981.
- [MS03] S. Mannor and N. Shimkin. The empirical Bayes envelope and regret minimization in competitive Markov decision processes. *Mathematics of Operations Research*, 28(2):327–345, May 2003.
- [Pat97] S.D. Patek. *Stochastic Shortest Path Games*. PhD thesis, LIDS, MIT, January 1997.
- [Put94] M. Puterman. *Markov Decision Processes*. Wiley-Interscience, 1994.
- [Sch93] A. Schwartz. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 298–305. Morgan Kaufmann, 1993.
- [Sin94] S. Singh. Reinforcement learning algorithms for average payoff Markovian decision processes. In *Proceedings of the 12th International Conference on Machine Learning*, pages 202–207. Morgan Kaufmann, 1994.

[Tsi94] J.N. Tsitsiklis. Asynchronous stochastic approximation and Q-learning. *Machine Learning*, 16:185–202, 1994.

A Appendix

In this appendix we provide convergence proofs of the two learning algorithms presented above. We start by from the Relative Q-learning algorithm and then turn to the λ -SSPG Q-learning algorithm. In both cases we also discuss the synchronous algorithm where it is assumed that all the state-action-action triplets are sampled simultaneously in every iteration. Much of the derivation here relies on [ABB01] and [BM00].

A.1 Proof of Theorem 1

We start with defining a synchronous version of (3.4).

$$Q_{n+1}(s, a, b) = Q_n(s, a, b) + \gamma(n) (r_\xi(s, a, b) + FQ_n(\xi(s, a, b)) - f(Q_n) - Q_n(s, a, b)) \quad (\text{A.7})$$

where $\xi(s, a, b)$ and $r_\xi(s, a, b)$ are the independently simulated random values of the next state and the immediate reward assuming $s_n = s$, $a_n = a$, and $b_n = b$, respectively. The above algorithm is the off-policy version of relative value iteration for average reward games.

Let us refer to Equation (A.7). In order to use the ODE method of [BM00] we first reformulate the synchronous Relative Q-learning iteration as a vector iterative equation:

$$Q_{n+1} = Q_n + \gamma(n)(TQ_n - f(Q_n)e - Q_n + M_{n+1}),$$

where: 1. TQ is the operator $T : \mathbb{R}^{S \cdot A \cdot B} \mapsto \mathbb{R}^{S \cdot A \cdot B}$ that is defined by: $TQ(s, a, b) = \sum_{s' \in \mathcal{S}} p(s'|s, a, b)(r(s, a, b) + FQ(s'))$; $f(Q)$ is a relative function as defined previously; and M_{n+1} is the “random” part of the iteration: $M_{n+1}(s, a, b) = r_\xi(s, a, b) + FQ(\xi(s, a, b)) - TQ_n(s, a, b)$. Denoting the σ -algebra until time n by $\mathcal{F}_n = \sigma(Q_m, M_m, m \leq n)$ it follows that for all n , under the assumption that all random variables are bounded: $\mathbb{E}(M_{n+1}|\mathcal{F}_n) = 0$ and $\mathbb{E}(\|M_{n+1}\|^2|\mathcal{F}_n) \leq C(1 + \|Q_n\|^2)$ for some constant C . We follow the analysis made by [ABB01] for the rest of the section. Lemmas whose proof is identical to the proof given in [ABB01] are stated without proof.

Let us define the following operators $T' : \mathbb{R}^{S \cdot A \cdot B} \rightarrow \mathbb{R}^{S \cdot A \cdot B}$ and $\hat{T} : \mathbb{R}^{S \cdot A \cdot B} \rightarrow \mathbb{R}^{S \cdot A \cdot B}$: $\hat{T}(Q) \triangleq TQ - ve$ and $T'(Q) \triangleq TQ - f(Q)e$, where v is the value of the game. In order to apply the ODE method we need to prove that the following ODE is asymptotically stable:

$$\dot{Q}(t) = T'(Q(t)) - Q(t). \quad (\text{A.8})$$

The operator T' is not a contraction, furthermore, it is not even non-expansive. We therefore establish its stability directly by considering the following ODE:

$$\dot{Q}(t) = \hat{T}(Q(t)) - Q(t). \quad (\text{A.9})$$

The following lemmas establish the properties of the operators.

Lemma 1. *The operator T is sup norm non-expansive*

Proof. Recall that $TQ(s, a, b) = \sum_{s' \in S} p(s'|s, a, b)(r(s, a, b) + FQ(s'))$. Fix Q_1 and Q_2 , the sup norm of the difference, $\|T(Q_1) - T(Q_2)\|_\infty$ is achieved by some element (s, a, b) .

$$\|T(Q_1) - T(Q_2)\|_\infty = \left| \sum_{s' \in S} p(s'|s, a, b) \left(\max_{v \in \Delta(A)} \min_{\mu \in \Delta(B)} \sum_{a', b'} Q_1(s', a', b') v_{a'}(s') \mu_{b'}(s') \right) - \max_{v \in \Delta(A)} \min_{\mu \in \Delta(B)} \sum_{a', b'} Q_2(s', a', b') v_{a'}(s') \mu_{b'}(s') \right|.$$

Assume without loss of generality that the sum inside the absolute value is positive. For every s' fix $v(s')$ which is a max-min strategy and first element is maximized (the element that relates to Q_1). Similarly, fix $\mu(s')$ for the second element which is a min-max strategy of P2 for each game defined by the second element for every s' . By the min-max theorem the first element cannot decrease and the second cannot increase. Since for every element s' the difference may only increase we have that:

$$\|T(Q_1) - T(Q_2)\|_\infty \leq \left| \sum_{s' \in S} p(s'|s, a, b) \left(\sum_{a'} \sum_{b'} (Q_1(s', a', b') - Q_2(s', a', b')) v_{a'}(s') \mu_{b'}(s') \right) \right|.$$

But this is a convex combination of elements of $Q_1 - Q_2$ and is certainly not more than the sup norm of the difference. \square

Corollary 1. *\hat{T} is sup norm non-expansive.*

Proof. $\|\hat{T}(Q_1) - \hat{T}(Q_2)\|_\infty = \|T(Q_1) - T(Q_2)\|_\infty \leq \|Q_1 - Q_2\|_\infty$ \square

Let us denote the span semi-norm by $\|\cdot\|_s$. That is $\|Q\|_s \triangleq \max_{s, a, b} Q(s, a, b) - \min_{s, a, b} Q(s, a, b)$.

Lemma 2. *The operator T is span semi-norm non-expansive.*

Proof.

$$\|TQ_1 - TQ_2\|_s = \max_{s, a, b} \{TQ_1(s, a, b) - TQ_2(s, a, b)\} - \min_{s', a', b'} \{TQ_1(s', a', b') - TQ_2(s', a', b')\}$$

There exist $(\bar{s}, \bar{a}, \bar{b})$ and $(\underline{s}, \underline{a}, \underline{b})$ that achieve the maximum and minimum of the span semi-norm, respectively. By writing the operator T explicitly and cancelling the reward elements:

$$= \sum_{s'} p(s'|\bar{s}, \bar{a}, \bar{b}) \left(\max_{v \in \Delta(A)} \min_{\mu \in \Delta(B)} \sum_{a', b'} Q_1(s', a', b') v_{a'} \mu_{b'} - \max_{v \in \Delta(A)} \min_{\mu \in \Delta(B)} \sum_{a', b'} Q_2(s', a', b') v_{a'} \mu_{b'} \right) - \sum_{s'} p(s'|\underline{s}, \underline{a}, \underline{b}) \left(\max_{v \in \Delta(A)} \min_{\mu \in \Delta(B)} \sum_{a', b'} Q_1(s', a', b') v_{a'} \mu_{b'} - \max_{v \in \Delta(A)} \min_{\mu \in \Delta(B)} \sum_{a', b'} Q_2(s', a', b') v_{a'} \mu_{b'} \right).$$

For every s there are four min-max operation in the above, lets us denote the maximizing strategy for P1's of the i -th item for state s by $v^i(s)$ and the minimizing strategy for P2's of the i -th item for state s by $\mu^i(s)$. For every s fix $v^1(s)$ as P1's strategy for the two first elements and $\mu^2(s)$ as P2's strategy for P2 strategy for the two first elements. The sum of the first two elements can only increase, as the first element cannot decrease and the second cannot increase. Similarly, for every s fix for the third and fourth elements, P1's strategy to be $v^4(s)$ and P2's strategy to be $\mu^3(s)$. The difference between the third and fourth elements can only increase, thus the total difference increases. We therefore obtain that $\|TQ_1 - TQ_2\|_s$ can be bounded by a convex combination of $Q_1 - Q_2$, which is certainly not greater than the span semi-norm. \square

Corollary 2. T' and \hat{T} are span semi-norm non-expansive.

Denote the set of equilibrium points of the ODE (A.9) by G , that is $G \triangleq \{Q : TQ = Q - ve\}$.

Lemma 3. G is of the form $Q^* + ce$.

Proof. First note that for every $c \in \mathbb{R}$ and $Q \in \mathbb{R}^{S \cdot A \cdot B}$ we have $T(Q + ce) = TQ + ce$ (e is now an SAB dimensional vector of ones). Also note that $F(Q + ce) = FQ + ce$ as equality in \mathbb{R}^S . Activate the operator F on the equation $TQ = Q - ve$, so that for $Q \in G$ we have that $FTQ = FQ - ce$. Under Assumption 1 we can apply Proposition 5.1 from [Pat97]. According to this proposition there exists a unique solution to the equation $TH = H - ve$ up to an additive constant. \square

Theorem 4. Q^* is the globally asymptotically stable equilibrium point for (A.8)

Proof. This is proved by direct computation using the above lemmas, and the Lipschitz continuity of f . We omit the details as the proof follows [ABB01, Theorem 3.4]. \square

We use the formulation of [BM00] for establishing convergence for the synchronous and the asynchronous cases. For the synchronous case we only need the stability assumption and the standard stochastic approximation assumption on the learning rate.

Theorem 5. Under Assumptions 1 and 2 the synchronous algorithm (A.7) converges to Q^* almost surely.

Proof. We apply Theorem 2.2 from [BM00] to show the boundedness of Q_n and to prove convergence to Q^* . As in [BM00], let $h(x) \triangleq T(Q) - Q - f(Q)e$. The ODE $\dot{x}(t) = h(x(t))$ has a globally stable solution by Theorem 4. Since $f(aQ) = af(Q)$ it follows that the limit $h_\infty \triangleq \lim_{z \rightarrow \infty} h(zx)/z$ exists and is simply the operator T with the payoffs $r(s, a, b)$ set to zero for all s, a, b . According to Theorem 4 the origin is asymptotically stable since the theorem can be applied to the game with zero payoffs. The other assumptions of Theorem 2.2 from [BM00] are satisfied by construction. \square

The asynchronous algorithm converges under the appropriate assumptions.

Proof of Theorem 1: This is a corollary of Theorem 2.5 in [BM00], the condition are satisfied as proved for the synchronous case. \square

A critical component in the proof is the boundedness of Q_n . We used the method of [BM00], however, one can show it directly as in [ABB01, Theorem 3.5]. By showing the boundedness directly a somewhat weaker assumption on f can be made, namely that $|f(Q)| \leq \|Q\|_\infty$ instead of $f(aQ) = af(Q)$.

A.2 Proof of Theorem 2

We associate with the average reward game an SSPG parameterized by $\lambda \in \mathbb{R}$. This SSPG has a similar state space, reward function, and conditional transition probability to the average reward game. The only difference is that s^* becomes an absorbing state with zero-reward, and the reward in all other states is reduced by λ . Let V_λ denote the value function of the associated SSPG which is given as the unique (by Proposition 4.1 from [Pat97]) solution of:

$$V_\lambda(s) = \text{val}_{a,b} \left[r(s, a, b) + \sum_{s' \in \mathcal{S}} p(s'|s, a, b) V_\lambda(s') - \lambda \right], \quad s \neq s^*, \quad (\text{A.10})$$

$$V_\lambda(s^*) = 0.$$

If $\lambda = v$ we retrieve the Bellman equation (2.2) for average reward. Let us first consider the synchronous version of (4.5):

$$Q_{n+1}(s, a, b) = Q_n(s, a, b) + \gamma(n) \left(r_\xi(s, a, b) + FQ_n((\xi_n(s, a, b))1_{\{\xi_n(s, a, b) \neq s^*\}} - \lambda_n - Q_n(s, a, b)), \right.$$

$$\lambda_{n+1} = \lambda_n + b(n)FQ_n(s^*), \quad (\text{A.11})$$

where we require that $b(n) = o(\gamma(n))$ and ξ and r_ξ are as before. The problem with using the ODE method directly is that λ_n may be unbounded. As in [ABB01], this can be solved using the projection method (e.g., [KY97]) by replacing the iteration of λ by: $\lambda_{n+1} = A[\lambda_n + b(n)FQ(s^*)]$, where $A(\cdot)$ is projection onto the interval $[-K, K]$ with K chosen so $v \in [-K, K]$. The following relies on a two time scale analysis as suggested in [Bor97]. The analysis closely follows Section 4 of [ABB01]. The limiting ODE of the iteration (A.10) assuming that $b(n) = o(\gamma(n))$ is:

$$\dot{Q}(t) = T'(Q(t), \lambda(t)) - Q(t), \quad \dot{\lambda}(t) = 0,$$

where $T'(Q, \lambda)$ is $T(Q) - \lambda e$. Thus, it suffices to prove that the following equation:

$$\dot{Q}(t) = T'(Q(t), \lambda) - Q(t) \quad (\text{A.12})$$

is asymptotically stable equation for a fixed λ . The stability can be deduced from the fact that T is a weighted maximum norm contraction as the following lemma proves. Recall that a weighted maximum norm with weights norm w in

\mathbb{R}^d is defined as: $\|x\|_w \triangleq \max_{1 \leq i \leq d} |x_i| w_i$. A policy is called *proper* in an SSPG if its total reward is finite (almost surely for every policy of the opponent). Assumption 1 implies that all policies are proper in the associated SSPG.

Lemma 4. *Assume that all the policies are proper in an SSPG. Then the operator $T(Q)$ is a weighted maximum norm contraction*

Proof. We define a Stochastic Shortest Path (SSP) problem where both players cooperate in trying to minimize the time of arrival to the absorbing state. Using the solution to this problem we bound the difference between Q-vectors when the players do not cooperate. Define a new single player SSP ([BT95], Section 2.3) where all the rewards are set to -1 (except for s^* which is zero reward) and the transition probabilities are unchanged. The two players are allowed to cooperate. By [BT95], there exists an optimal reward \tilde{J} and stationary policies $\tilde{\mu} \in \Delta(\mathcal{A})^S$ for P1 and $\tilde{\nu} \in \Delta(\mathcal{B})^S$ for P2 such that the optimal time of arrival to the absorbing state is minimal. The vector \tilde{Q} is defined as: $\tilde{Q}(s, a, b) = \sum_{s'} p(s'|s, a, b) \tilde{J}(s')$, and Bellman's equation for that SSP is: $\tilde{Q}(s, a, b) = -1 + \sum_{s'} p(s'|s, a, b) \sum_{a', b'} \tilde{\mu}_{a'}(s') \tilde{\nu}_{b'}(s') \tilde{Q}(s', a', b')$. Moreover, for any μ and ν we have $(P(s'|s, \mu, \nu))$ is the transition matrix assuming μ and ν are played), in vector notations: $\tilde{Q} \leq -1e + P(s'|s, \mu, \nu)\tilde{Q}$. that is:

$$-\sum_{s'} p(s'|s, a', b') \sum_{a', b'} \mu_{a'}(s') \nu_{b'}(s') \tilde{Q}(s', a', b') \leq -\tilde{Q} - 1 \leq \tilde{Q}(s, a, b)\alpha, \quad (\text{A.13})$$

where $\alpha \triangleq \max_{s, a, b} (\tilde{Q}(s, a, b) + 1) / (\tilde{Q}(s, a, b))$, since $\tilde{Q} \leq -1$ we have $\alpha \in [0, 1)$. We now show that α is the contraction factor for the weighted max norm which vector is $w \triangleq -\tilde{Q}$.

Resume the discussion of the original SSPG, let Q and \bar{Q} be elements such that $\|Q - \bar{Q}\|_w^\infty = c$. Let $\mu \in \Delta(\mathcal{A})^S$ be a policy such that $T_\mu Q = TQ$ (maximizing policy), where:

$$T_\mu Q(s, a, b) \triangleq \sum_{s'} p(s'|s, a, b) \left(r(s, a, b) + \min_{u \in \Delta(\mathcal{B})} \sum_{a', b'} \mu_{a'}(s') u_{b'}(s') Q(s', a', b') \right).$$

Let $\nu \in \Delta(\mathcal{B})^S$ be a policy for P2 such that $T_\mu \bar{Q} = T_{\mu\nu}(\bar{Q})$ (minimizing policy for P2) where $T_{\mu\nu}(s, a, b) = \sum_{s'} p(s'|s, a, b) (r(s, a, b) + \sum_{a', b'} \mu_{a'}(s') \nu_{b'}(s') Q(s', a', b'))$. It follows then: $TQ - T\bar{Q} = T_\mu Q - T\bar{Q} \leq T_\mu Q - T_\mu \bar{Q} = T_\mu Q - T_{\mu\nu} \bar{Q} \leq T_{\mu\nu} Q - T_{\mu\nu} \bar{Q}$. The inequalities follow by imposing on the minimizer and the maximizer policies that might be suboptimal. We therefore have that for every s, a, b

$$\frac{TQ(s, a, b) - T\bar{Q}(s, a, b)}{cw(s, a, b)} \leq \frac{1}{cw(s, a, b)} \sum_{s'} \sum_{a', b'} p(s'|s, a', b') \mu_{a'}(s') \nu_{b'}(s') (Q(s', a', b') - \bar{Q}(s', a', b')),$$

since $\|Q - \bar{Q}\|_w^\infty = c$ we have $Q - \bar{Q} \leq cw$ (as a vector inequality) and therefore:

$$\frac{TQ(s, a, b) - T\bar{Q}(s, a, b)}{cw(s, a, b)} \leq \frac{1}{w(s, a, b)} \sum_{s'} \sum_{a', b'} (p(s'|s, a', b') \mu_{a'}(s') \nu_{b'}(s') w(s', a', b')).$$

Plugging in w as defined above:

$$\frac{TQ(s, a, b) - T\bar{Q}(s, a, b)}{cw(s, a, b)} \leq \frac{1}{-\bar{Q}(s, a, b)} \sum_{s'} \sum_{a', b'} p(s'|s, a', b') \mu_{a'}(s') \nu_{b'}(s') (-\bar{Q}(s', a', b')).$$

Finally, using the previous argument regarding the minimality of $\tilde{\mu}$ and $\tilde{\nu}$ and (A.13) we have

$$\frac{TQ(s, a, b) - T\bar{Q}(s, a, b)}{cw(s, a, b)} \leq \frac{1}{-\bar{Q}(s, a, b)} (-\bar{Q}(s, a, b)) \alpha = \alpha < 1.$$

□

Let $Q^*(\lambda)$ be the Q -vector that appears in each entry of (A.10). Adapting the arguments of [BS99] and using the fact that $T'(\cdot, \lambda)$ is a weighted maximum norm contraction we can deduce that:

Lemma 5. *The globally asymptotically stable equilibrium for (A.12) is $Q^*(\lambda)$. Furthermore, every solution of the ODE (A.12) satisfies that $\|Q(t) - Q^*(\lambda)\|_w \rightarrow 0$ monotonically.*

In order to use two time scale stochastic approximation (e.g., [Bor97]) convergence theorem we need to establish the boundedness of Q :

Lemma 6. *Q_n remains bounded almost surely for both the synchronous (A.11) and asynchronous (4.5) iterations.*

Proof. According to Lemma 4 we have: $\|T(Q)\|_w \leq \alpha\|Q\|_w + D$. Since λ is bounded by K there exists some D such that $\|T'(Q, \lambda)\|_w \leq \alpha\|Q\|_w + D + K$. If $\|Q\|_w \geq 2/(1-\alpha)(D+K)$ we have $\|TQ\|_w \leq \alpha\|Q\|_w + D + K \leq (1/2 + \alpha/2)\|Q\|_w$ and therefore for Q whose norm is large enough the iteration reduces the norm. The asynchronous case follows in a similar manner to [BT95] (Section 2.3). □

A convergence theorem can finally be proved in a similar manner to [ABB01].

Theorem 6. *Suppose that Assumptions 1 and 2 hold. Then the synchronous λ -SSPG Q -learning algorithm (A.11) satisfies that $(Q_n, \lambda_n) \rightarrow (Q^*, v)$ almost surely.*

Proof. The assumptions needed for Theorem 1.1 in [Bor97] are satisfied by construction. By definition λ_n is bounded. The vector Q_n is bounded by Lemma 6. Since T' is continuous w.r.t. λ and using the stability of the underlying ODE (Lemma 5) we have ensured convergence to the appropriate limit. The only difference from Theorem 4.5 in [ABB01] is that we need to make sure that the slope of the mapping $\lambda \rightarrow Q^*(\lambda)$ is finite. But this was shown by Lemma 5.1 of [Pat97]. □

For the asynchronous case the same can be proved.

Proof of Theorem 2: The analysis of [Bor98, KB00] applies since boundedness holds by Lemma 6. The only difference from Theorem 6 is that a time scaled version is used. □