McGill University

ECSE 506: Term Project

Q-Learning for Markov Decision Processes*

Authors:

Khoa Phan
khoa.phan@mail.mcgill.ca

Sandeep Manjanna
sandeep.manjanna@mail.mcgill.ca

*(*Based on: "Convergence of One-step Q-learning", an extract from Chris Watkins' PhD Thesis*
and
"Asynchronous Stochastic Approximation and Q-Learning" by John N. Tsitsiklis)

Contents

1 Introduction

This report summarizes two major works in the field of Q-Learning by Christopher Watkins and John N Tsitsiklis. Q-Learning is a reinforcement learning technique that works by learning an action-value function that gives the expected utility of taking a given action in a given state and following a fixed policy thereafter. The report first starts with a brief introduction to the filed of reinforcement learning along with an algorithm for Q-learning. Next section introduces the problem addressed in the report followed by the convergence proof of Watkins [1]. This is followed by the advanced proofs and further work continued by Tsitsiklis [2]. Finally, the report provides the comparisons between these two works along with the improvements and further studies done in the field of Q-learning.

1.1 Reinforcement learning

Reinforcement learning is an area of machine learning, concerned with how an agent ought to take actions in an environment so as to maximize some notion of cumulative reward. Reinforcement learning is learning what to do–how to map situations to actions–so as to maximize a numerical reward signal. The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards [4]. As an example, consider the dog being taught to fetch the ball. The dog is awarded with a cookie (positive reward) or with scolding (negative reward) based on its action of fetching the ball or not. Thus dog learns to fetch the ball as many times as it wants the cookie. It gained the knowledge of how to fetch the ball by its experience and the drive to do so was the positive reward.

1.2 Q-Learning

Q-learning is a reinforcement learning technique that works by learning an action-value function that gives the expected utility of taking a given action in a given state and following a fixed policy thereafter [4]. One of the strengths of Q-learning is that it is able to compare the expected utility of the available actions without requiring a model of the environment. In one-step Q-learning, the Q-value updation can be expressed as below [3]:

$$Q_n(x,a) = \begin{cases} (1-\alpha_n)Q_{n-1}(x,a) + \alpha_n[r_n + \gamma V_{n-1}(y_n)], & x = x_n \text{ and } a = a_n \\ Q_{n-1}(x,a) & \text{otherwise} \end{cases}$$

where $V_n(.)$ is the updated value function in state n.

1.3 Problems to be addressed

i) Convergence of Q-Learning:

This is an extract from Watkins' work in his PhD Thesis. In his work, the convergence is proved by constructing a notional Markov decision process called Action- Replay Process, which is similar to the real process. Then it is shown that the Q-Values produced by the one step Q-learning process after 'n' training examples, are the exact Optimal action values for the start of the action-replay process for 'n' training examples.

The work [1] also continues to prove that, as more data is used, the optimal action-value function at the start of action-replay process converges to the optimal action-value function of the real process.

ii) Extensions to convergence theorem and Q-learning:

In his work, Tsitsiklis considers both discounted and un-discounted MDPs without assuming all policies must lead to a zero-cost absorbing state. There are a set of different assumptions and theorems proposed by Tsitsiklis in his work [2]. And here, the actual process is modeled and not the Action-Replay Process as done in the work by Watkins.

2 Convergence of Q-Learning

Convergence of Q-Learning the way of proving that the learned action-value function, Q, directly approximates Q*, the optimal action-value function, independent of the policy being followed. Here, proving the convergence for Q-Learning can be stated to be same as proving that $Q_n(x, a) \to Q^*(x, a)$ as $n \to \infty$. The key to the convergence proof is an artificial controlled Markov process called the Action-Replay Process (ARP), which is constructed from the episode sequence and the learning rate sequence α_n.

2.1 Action Replay Process (ARP)

The ARP is a purely notional Markov decision process, which is used as a proof device. This process is constructed progressively from the sequence of observations. There is one better way of illustrating the ARP mentioned in one of the technical notes. According to that explanation, an ARP is described as a card game, where each episode $< x_t, a_t, y_t, r_t, \alpha_t >$ is written on the card. Here, x_t is the current state of the process at t,a_t is the action taken at time t, y_t is the resulting state when action a_t is executed in state x_t, r_t is the immediate reward of action a_t at state x_t and α_t is the learning rate of the state $< x_t, t >$. At the layer

k, i.e. equivalent to say at card k, the state of the system is $< x, k >$ and it corresponds to each state x in the Real Process. The bottom card (numbered 0) has written on it the agent's initial values $Q_0(x, a)$ for all pairs of x and a.

The next state of the ARP, given the current state (x, n) and action a, is determined according to the following process. First, all the cards for episodes later than n are eliminated, leaving just a finite deck. Cards are then removed one at a time from top of this deck and examined until one is found whose starting state and action match x and a, say at episode t. Then a biased coin is flipped, with a probability at α_t of coming out heads, and $(1 - \alpha_t)$ of tails. If the coin turns up heads, the episode recorded on this card is replayed, a process described below; if the coin turns up tails, this card too is thrown away and the search continues for another card matching x and a.

If the bottom card is reached, the game stops in a special, absorbing, state, and just provides the reward written on this card for x, a, namely $Q_0(x, a)$. Replaying the episode on card t consists of emitting the reward r_t, written on the card t and then moving to the next state $< y_t, t - 1 >$ in the ARP, where y_t is the state real process went on the episode t. The next state transition of the ARP will be taken based on just the remaining deck. No matter what actions are chosen, if one starts at level k, each action will lead to a new state at lower level, until finally one reaches the level 0 and the process terminates. Thus it is sure that the ARP terminates. Below is the algorithm [1] for performing action a in the state $< x, k >$,

```
To perform $a$ in $< x, 0 >$,
              terminate the ARP with the immediate reward of $Q_0(x, a)$,
              and halt.
To perform $a$ in $< x, k >$ for $k > 0$,
              if $x = x_n$ and $a = a_n$
                        then begin
                              either ( with probability $\alpha_k$)
                                     go to $< y_k, k - 1 >$ with an immediate reward of $r_k$
                                     and halt,
                              or ( with probability 1-$\alpha_k$)
                                 perform $a$ in $< x, k - 1 >$
                        end
              else
                  perform $a$ in $< x, k - 1 >$.
```

Watkins also makes a point in the work that the episodes need not form a continuous sequence, that is the y of one episode need not be the x of the next episode. Hence, this makes the system non real-time. Thus this approach cannot be applied for on-line learning. The further proof of convergence for on-line Q-Learning is provided by Tsitsiklis in his work.

2.2 Action - Replay Theorem

The aim of this theorem is to prove that for all states x, actions a and stage n of ARP, $Q_n(x,a) = Q^*_{ARP}(< x, n >, a)$. The proof for this theorem is given by Watkins is through induction. Lets consider the initial state at level 0, $Q_0(x,a)$ is the optimal, indeed the only possible Q-value of $< x, 0 >$,a. Therefore,

$$Q_0(x,a) = Q^*_{ARP}(< x, 0 >, a)$$

Hence, the theorem holds for $n = 0$.
Now, lets assume that the Q-values Q_{n-1}, generated by Q-learning rule, are optimal Q-values for ARP at stage $n-1$, that is

$$Q_{n-1}(x,a) = Q^*_{ARP}(< x, n-1 >, a)$$

This implies that $V_{n-1}(x)$ are the optimal values for $n - 1^{th}$ stage, that is

$$V^*_{ARP}(< x, n-1 >) = \max_a Q_{n-1}(x,a)$$

According to the Q-value updation equation,

$$Q_n(x,a) = \begin{cases} (1-\alpha_n)Q_{n-1}(x,a) + \alpha_n[r_n + \gamma V_{n-1}(y_n)] & if x = x_n, a = a_n \\ Q_{n-1}(x,a) & \text{otherwise} \end{cases}$$

for all x, a not equal to x_n, a_n, performing a in $< x, n >$ in ARP gives exactly same results as performing a in $< x, n-1 >$. Therefore we have,

$$Q^*_{ARP}(< x, n >, a) = Q^*_{ARP}(< x, n-1 >, a)$$

Therefore for all x,a not equal to x_n,a_n respectively, $Q_n(x,a) = Q^*_{ARP}(< x, n >, a)$

Now, considering the other case where action a_n is performed on the ARP state $< x_n, n >$. The optimal action value in the ARP of $< x_n, a_n >$ is,

$$Q^*_{ARP}(< x_n, n >, a_n) = \alpha_n(r_n + \gamma V^*_{ARP}(< y_n, n-1)) + (1-\alpha_n)Q^*_{ARP}(< x_n, n-1 >, a_n)$$

$$= \alpha_n(r_n + \gamma V_{n-1}(y_n)) + (1-\alpha_n)Q_{n-1}(x_n, a_n)$$

$$= Q_n(x_n, a_n)$$

Hence proved by induction that $Q_n(x,a) = Q^*_{ARP}(< x, n >, a)$ for all x, a, and $n \geq 0$.

2.3 Convergence of Q^*_{ARP} to Q^*

In this section, we discover what are the conditions under which Watkins proposes the convergence of optimal action values for the action replay process at n^{th} stage to that for the real precess as $n \to \infty$
The following are the assumptions made by Watkins in his work,

- There are infinite number of observations ($n \to \infty$).
- The learning rate α_n for observations are positive and decreasing monotonically With increasing n. i.e $\alpha_n \to$ as $n \to \infty$.
- The sum of the learning rates α_n for observations is infinite. i.e.

  $$\sum_{n=1}^{\infty} \alpha_n = \infty \text{ and } \sum_{n=1}^{\infty} \alpha_n^2 < \infty$$

  The method used by Watkins to demonstrate that these assumptions are sufficient is to show that if one starts from the $n^t h$ layer of the replay process, then the replay process will approximate the real process to any given degree of accuracy for any given finite number of stages, provided that n is chosen to be large enough.

  Watkins considers the concept of depth of a state-action pair $d(< x, k >, a)$ in the replay process to be the sum of the learning factors for all observations of the form $[xar_l y_l]$ with $l \leq k$. Thus the probability of reaching $< x, 0 >$ becomes arbitrarily small as depth, $d(< x, k >, a)$, becomes large. This is because, as we go on to the higher stages in the ARP, the chances of finding the same set of state-action pair as that of the current stage becomes high and the need to iterate till we reach the end of the layer, i.e. reaching 0-level, becomes very less. Thus if we choose sufficiently high value for n, then the chances of reaching the bottom level becomes zero.

  The $d(< x, k >, a) \to \infty$ as $n \to \infty$. This is followed by the third assumption made above. For any given D, and any given ε, it is possible to choose n such that,

  $$\max_{m>n}\{\alpha_m\} < \varepsilon$$

  For any such chosen n, it is then possible to choose n' such that,

  $$\min_{x,a}\{d(< x, n' >, a) - d(< x, n >, a)\} > D$$

  Thus based on the above two conditions, it is possible to create a sequence of layers $(n_1, n_2, n_3.....)$ such that, every layer has a depth of D between each of them. i.e. there is at least a depth of D between the layer n_1 and layer n_2, layer n_2 and layer n_3 and so on. It is therefore possible to chose an n so large that the minimum possible number of replayed observations is larger than any chosen k with a probability as close to one as desired.

It is also possible to choose a n so large that, the maximum learning rate α is so small that the transition probabilities and reward means of the ARP are uniformly as close as desired to the transition probabilities and reward means of the Real Process, with a probability as close to 1 as desired. Thus when we choose a very large n, the value of α becomes negligible and thus making the ARP tend toward the Real Process.

Thus, when we choose an n so large that Q^*_{ARP} at the n^{th} level of ARP is uniformly close to the corresponding optimal action values of the Real Process and thus the proof for convergence of Q^*_{ARP} to Q^*. Thus we can say that, under the assumed condition, the action value of the Q-learning process converges with a probability of 1 as $n \to \infty$.

3 Stochastic Approximation and Q-Learning

3.1 What has Tsitsiklis done on top of Watkins's work ?

In Q-learning, transition probabilities and costs are unknown but information on them is obtained either by simulation or by experimenting with the system to be controlled. As a recursive algorithm, in each stage, the Q-learning uses new information to compute an additive correction term to the old estimates. Since these correction terms are random, Q-learning has the same general structure as stochastic approximation algorithms. Hence, by combining ideas from stochastic approximation theory and convergence theory of parallel asynchronous algorithms, the work [2] has established the convergence of Q-learning under various settings.

Besides a new proof for the results of [1], the author has extended convergence proof of Q-learning in several directions.

– The convergence for undiscounted problems without assuming that all policies must lead to a zero-cost absorbing state is shown. In [1], the author assumed all policies lead to a zero-state absorbing state, i.e., proper policies. In [2], the technicality requires that there exists at least one proper stationary policy and every improper policies yield infinite expected cost for at least one initial state. Also, the updated Q values need to be bounded.
– The costs per stage is allowed to be unbounded random variables with finite variance. In [1], the cost (or reward) must be bounded random variable.
– The convergence of Q-learning for the case of parallel implementation allowing the use of outdated information is also established. Synchronous updates are assumed in [1].

– Regard the implementation, the action-replay process in [1] can be implemented offline while the direct Q-learning algorithms in [2] is done online.

These convergence results in [2] do not follow from the available convergence theory of stochastic approximation algorithms. Hence, the author first extended the classical results of stochastic approximation theory and then showed that various settings of Q-learning algorithms are special cases of these new results.

In this report, for simplicity, we mainly review results in [2] for the case of discounted cost and synchronous updates only. The results on undiscounted costs are discussed toward the end.

3.2 Stochastic approximation: Assumptions and results

The following algorithm consists of noisy updates for a vector $x \in \mathcal{R}^n$ to find fixed-point solution of the equation $F(x) = x$ where $F(x) = (F_1(x), \ldots, F_n(x))$ for all $x \in \mathcal{R}^n$.

Let $\mathcal{N}$ be the set of nonnegative integers. We employ a discrete 'time' variable t, taking values in $\mathcal{N}$. This variable need not have any relation with real time; rather, it is used to index successive updates. Let $x(t)$ be the value of the vector x at time t and let $x_i(t)$ denote its i-th component. Let T^i be an infinite subset of $\mathcal{N}$ indicating the set of times at which an update of x_i is performed. We assume that:

$$x_i(t+1) = x_i(t), \quad t \notin T^i \tag{1}$$

When x_i is updated at times in T^i, the update equation is of the following form:

$$x_i(t+1) = x_i(t) + \alpha_i(t)(F_i(x(t)) - x_i(t) + w_i(t)), \quad t \in T^i \tag{2}$$

Here, $\alpha_i(t)$ is a stepsize parameter belonging to $[0, 1]$, $w_i(t)$ is a noise term. To unify (1) and (2), it is convenient to assume that $\alpha_i(t) = 0$ for $t \notin T^i$. Since the set T^i is infinite, each component is updated infinitely many times.

Since T^i is random in general (which can also be assumed deterministic), the variables introduced so far $(x(t), \alpha_i(t), w_i(t))$ are viewed as random variables defined on a probability space $(\Omega, \mathcal{F}, \mathcal{P})$ and the assumptions deal primarily with the dependencies between these random variables. Our assumptions also involve an increasing sequence $\{\mathcal{F}\}_{t=0}^{\infty}$ of subfields of $\mathcal{F}$ and $\mathcal{F}(t)$ is meant to represent the history of the algorithm up to, and including the point at which the stepsizes $\alpha_i(t)$ for the t-th iteration are selected, but just before the noise term $w_i(t)$ is generated.

The following assumptions impose on the statistics of the random variables involved in the algorithm.

Assumption 1:

a) $x(0)$ is $\mathcal{F}(0)$-measurable;

b) For every i and t, $w_i(t)$ is $\mathcal{F}(t+1)$-measurable;

c) For every i and t, $\alpha_i(t)$ is $\mathcal{F}(t)$-measurable;

d) For every i and t, $E\Big[w_i(t)|\mathcal{F}(t)\Big] = 0$;

e) There exists constants A and B such that

$$E\Big[w_i^2(t)|\mathcal{F}(t)\Big] \leq A + B \max_k \max_{\tau \leq t} |x_k(\tau)|^2$$

for all i, t.

Assumption 2:

a) For every i, $\sum_{t=0}^{\infty} \alpha_i(t) = \infty$;

b) There exists constants C such that for every i, $\sum_{t=0}^{\infty} \alpha_i^2(t) \leq C$.

Assumption 3: There exists a vector $x^* \in \mathcal{R}^n$, a positive vector v, and a scalar $\beta \in [0,1)$, such that $\|F(x) - x^*\|_v \leq \beta \|x - x^*\|_v$ for all $x \in \mathcal{R}^n$ where $\|x\|_v = \max_i \frac{|x_i|}{v_i}$ for $x \in \mathcal{R}^n$.
When all components of v are equal to 1, $\|.\|_v$ is the same as the maximum norm $\|.\|_\infty$.

The following result regarding the convergence of $x(t)$ is proved in [2].

Theorem 1 *Let Assumptions 1, 2, 3 hold. Then, $x(t)$ converges to x^* with probability 1.*

PROOF: See [2] for details.

Next, the author showed that the Q-learning algorithm can be viewed as special case of stochastic approximation algorithm.

3.3 Q-learning and stochastic approximation

Consider a Markov decision problem defined on the a finite state space S. For every $i \in S$, there is a finite set $U(i)$ of possible actions. A set of nonnegative scalars, or transition probabilities, $p_{ij}(u), u \in U(i), j \in S$ such that $\sum_{j \in S} p_{ij}(u) = 1$ for all $u \in U(i)$. For every state i and control action u, the one-stage cost c_{iu} is random variable with finite variance. A stationary policy is a function π defined on S such that $\pi(i) \in U(i)$ for all $i \in S$. A policy π induces a Markov chain with transition probabilities:

$$Pr(s^\pi(t+1) = j | s^\pi(t) = i) = p_{ij}(\pi(i)).$$

Given a stationary policy π and an initial state $i \in S$, the cost-to-go V_i^π is defined:

$$V_i^\pi = \limsup_{T\to\infty} E\Big[\sum_{t=0}^{\infty} \beta^t c_{s^\pi(t),\pi(s^\pi(t))} | s^\pi(0) = i\Big]$$

where $\beta \in (0,1]$ is discount factor. We assume $\beta < 1$ for now. The optimal cost-to-go is thus:

$$V_i^* = \inf_\pi \; V_i^\pi.$$

Dynamic programming operator $T : \mathcal{R}^{|S|} \to \mathcal{R}^{|S|}$ with components T_i is defined as:

$$T_i(V) = \min_{u\in U(i)} \left\{ E[c_{iu}] + \beta \sum_{j\in S} p_{ij}(u) V_j \right\}.$$

T contraction w.r.t. norm $\|\|_\infty$ and V^* unique fixed point.

Q-learning is a method to compute V^* based on a reformulation of the Bellman equation $V^* = T(V^*)$. In particular, let $P = \Big\{(i,u) | i \in S, u \in U(i)\Big\}$ set of all possible state-action pairs and let cardinality $|P| = n$. After t iterations, vector $Q(t) \in \mathcal{R}^n$ with components $Q_{iu}(t), (i,u) \in P$ are updated as:

$$Q_{iu}(t+1) = Q_{iu}(t) + \alpha_{iu}(t)\Big[c_{iu} + \beta \min_{v\in U(s(i,u))} Q_{s(i,u),v}(t) - Q_{iu}(t)\Big] \tag{3}$$

$\alpha_{iu}(t)$ is a nonnegative stepsize coefficient which is set to 0 for those $(i,u) \in P$ for which Q_{iu} is not updated in the current iteration. c_{iu} is random cost and $s(i,u)$ is random successor state which is equal to j with probability $p_{ij}(u)$. All random samples are drawn independently.

The Q-learning algorithm (3) is shown to have the form of a stochastic approximation algorithm. Let F be the mapping from $\mathcal{R}^n$ to itself with components F_{iu} defined as

$$F_{iu}(Q) = E[c_{iu}] + \beta E\Big[\min_{v\in U(s(i,u))} Q_{s(i,u),v}\Big]$$

Note that

$$E\Big[\min_{v\in U(s(i,u))} Q_{s(i,u),v}\Big] = \sum_{j\in S} p_{ij}(u) \min_{v\in U(j)} Q_{jv}.$$

It can be seen that if Q fixed point of F then vector V with $V_i = \min_{u\in U(i)} Q_{iu}$ fixed point point of T. Rewrite the Q-learning equation:

$$Q_{iu}(t+1) = Q_{iu}(t) + \alpha_{iu}(t)\Big(F_{iu}(Q(t)) - Q_{iu}(t) + w_{iu}(t)\Big)$$

where the noise term $w_{iu}(t)$ is:

$$w_{iu}(t) = c_{iu} - E[c_{iu}] + \min_{v\in U(s(i,u))} Q_{s(i,u),v}(t) - E\Big[\min_{v\in U(s(i,u))} Q_{s(i,u),v}(t) | \mathcal{F}(t)\Big]. \tag{4}$$

3.3.1 Q-learning satisfies the Assumptions 1, 2, 3

We can see from (4) that $E[w_{iu}(t)|\mathcal{F}(t)] = 0$. The conditional variance of $\min_{v \in U(s(i,u))} Q_{s(i,u),v}$ given $\mathcal{F}(t)$ is bounded above by $\max_{j \in S} \max_{v \in U(j)} Q^2_{jv}(t)$, and hence:

$$E[w^2_{iu}(t)|\mathcal{F}(t)] \leq \text{Var}(c_{iu}) + \max_{j \in S} \max_{v \in U(j)} Q^2_{jv}(t)$$

Hence, Q-learning satisfies assumption 1. Assumption 2 is satisfied by imposing the conditions on the stepsizes $\alpha_{iu}(t)$ for all $(i, u) \in P$ and t. Also, it requires that every state-action pair (i, u) is simulated an infinite number of times.

By definition of the mapping F, we have:

$$|F_{iu}(Q) - F_{iu}(Q')|_\infty \leq \beta \max_{j \in S, v \in U(j)} |Q_{jv} - Q_{jv}|, \quad \forall Q, Q'.$$

Hence, F is contraction mapping w.r.t. maximum norm $\|.\|_\infty$. Hence, assumption 3 is satisfied.

The convergence of Q-learning is established by Theorem 1.

3.3.2 Undiscounted Q-learning

Now consider Q-learning for undiscounted case $\beta = 1$. It is then assumed that there is a cost-free state, say state 1, which is absorbing; that is, $p_{11}(u) = 1$ and $c_{1u} = 0$ for all $u \in U(1)$. We say that a stationary policy is proper if the probability of being at the absorbing state converges to 1 as time converges to infinity; otherwise, we say that the policy is improper.

Assumption 4

a) There exists at least one proper stationary policy.
b) Every improper stationary policy yields infinite expected cost for at least one initial state.

In [1], the author assumed that all policies are proper which is stronger than that in Assumption 4. When $\beta = 1$, the mapping T is not, in general, a contraction. However, it is still true that the set $\left\{ V \in \mathcal{R}^{|S|} \mid V_1 = 0 \right\}$ contains a unique fixed point of T and this fixed point is equal to V^*, as long as Assumption 4 holds.

Under Assumption 4, [2] showed that the convergence of Q-learning also follows from Theorem 1.

4 Conclusions

This report has reviewed the convergence proofs for Q-learning for Markov decision processes under various settings. In [1], the author has proposed an action-replay process which simulates a real Q-learning process. The simulated process is shown to be convergent which implies the convergence of Q-learning. [2], the author proved the convergence of Q-learning algorithms by extending the results in stochastic approximation theory. [2] has extended the results in [1].

Further work is done by Barto, Bradtke Singh, on updating multiple Q-values in every iteration. Also lot of work is done on convergence for un-discounted models of Q-learning.

References

[1] Watkins, *Learning from Delayed Rewards.* Doctoral dissertation. University of Cambridge, Cambridge, United Kingdom, 1989.

[2] J. N. Tsitsiklis, "Asynchronous Stochastic Approximation and Q-learning," *Machine Learning*, no. 16, pp. 185–202, 1994.

[3] Christopher J.C.H. Watkins and Peter Dayan, *Techmical Note Q-Learning.* 1992

[4] Richard S. Sutton and Andrew G. Barto, "Reinforcement Learning: An Introduction",*A Bradford Book*, The MIT Press, Cambridge, Massachusetts, London, England.