ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

$\mathcal{QL}_2$, a Simple Reinforcement Learning Scheme for Two-Player Zero-Sum Markov Games

Benoît Frénay[1] and Marco Saerens[2] *

Université catholique de Louvain - Machine Learning Group
1- FSA/ELEC/DICE - Place du Levant, 3 1348 Louvain-la-Neuve, Belgium
2- ESPO/LSM/ISYS - Place des Doyens, 1 1348 Louvain-La-Neuve, Belgium

Abstract. Markov games are a framework which formalises n-agent reinforcement learning. For instance, Littman proposed the minimax-Q algorithm to model two-agent zero-sum problems. This paper proposes a new simple algorithm in this framework, $\mathcal{QL}_2$, and compares it to several standard algorithms (Q-learning, Minimax and minimax-Q). Experiments show that $\mathcal{QL}_2$ converges to optimal mixed policies, as minimax-Q, while using a surprisingly simple and cheap gradient-based updating rule.

1 Introduction

Reinforcement learning (RL, see e.g. [1, 2, 3] for theory and [4] for applications) allows modeling and solving problems for which it is impossible to obtain a learning set: the only available information is a numerical reward received upon success (partial or not). With algorithms such as Q-learning [5], the agent learns the actions that lead to the rewards from direct interaction with its environment.

In this paper, we address specifically two-player zero-sum games (Sec. 2) described in Littman's paper [6] which (i) compares three different models (MDP's, matrix games [7] and Markov games), (ii) proposes the minimax-Q algorithm, which exploits the links between RL and game theory and (iii) demonstrates the respective strengths of Q-learning and minimax-Q on a soccer game.

Littman's minimax-Q is an efficient, optimal, algorithm, but it has to solve a linear programming (LP) problem at each step. We propose a new gradient-based algorithm $\mathcal{QL}_2$ (Sec. 3) which also experimentally achieves optimality, but is much simpler and avoids the computational overhead due to a LP solver. We compare experimentally minimax-Q and $\mathcal{QL}_2$(Sec. 4) with several algorithms implemented with the `Qash` library [8] on Littman's soccer game [6].

The interest of $\mathcal{QL}_2$ is its cheap policy update rule: whereas Littman's algorithm solves a LP problem, ours involves only arithmetical operations.

2 Basic Framework and State of the Art Algorithms

In this paper, we consider an **agent** $\mathcal{A}g$ interacting with a discrete **environment** $\mathcal{E}$ in competition with an **opponent** $\mathcal{O}pp$. $\mathcal{A}g$ performs actions, perceives $\mathcal{O}pp$'s actions and $\mathcal{E}$'s **state** X_t which belongs to its set of states $\mathcal{S}$ and aims to learn an optimal policy with no prior knowledge. Its task is specified by a **reward signal**

*Thanks to Michel Verleysen for its advices along the writing of this paper.

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

r and a **return** R. r indicates the desirability of each state-actions tuple and can be compared to fear and pleasure in ethology. R quantifies the accumulated amount of rewards: in this paper, we use the **infinite-horizon discounted return** [1, 3] $R_t = \sum_{i=0}^{+\infty} \gamma^i r_{t+i}$ where $\gamma \in]0, 1[$ is the **discount rate**.

2.1 Markov Games

In this work, we suppose that the interest of both agents is **opposite**. In such a case, the above two-agent environment can be modeled as a **two-player zero-sum Markov game** $\mathcal{MG}$ [6]. At each time step t, $\mathcal{A}g$ and $\mathcal{O}pp$ choose their actions a and o among the available actions $A(s)$ and $O(s)$ which belong to their actions sets $\mathcal{A}$ and $\mathcal{O}$. $\mathcal{MG}$'s next state depends on the transition probability

$$T_{s_i,a,o,s_j} = \Pr(X_{t+1} = s_j | X_t = s_i, a, o); \tag{1}$$

$\mathcal{A}g$ and $\mathcal{O}pp$ respectively receive $r_s(a, o)$ and $-r_s(a, o)$ in s. In this paper, $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{O}$ are finite and $\mathcal{MG}$ has an unique initial state s_{init} such that $T_{s,a,o,s_{init}} = 0$.

$\mathcal{A}g$ follows a **policy** π (see [6, 9]) which can be either pure, if it associates to each state s the action $\pi(s)$, or mixed, if it associates to each state s and action $a \in A(s)$ the probability $\pi_s(a)$ to choose a. $\mathcal{O}pp$'s policy is denoted σ.

2.2 A Template for Online Markov Games Algorithms

Game theory [7] tells us that $\mathcal{O}pp$, if rational, has to minimise $\mathcal{A}g$'s **expected return**. Therefore, if we define the **value function** V_π such that

$$V_\pi(s) = \min_\sigma \mathop{\mathrm{E}}_{\pi,\sigma} \left\{ \sum_{i=0}^{+\infty} \gamma^i r_{t+i} \middle| X_t = s \right\} \tag{2}$$

then the corresponding (not necessarily unique) **optimal policy** π^* maximises $V_\pi(s), \forall s \in S$. When T and r are unknown, the learning is performed online with temporal difference algorithms [1, 2] using the **evaluation function** Q_π

$$Q_\pi(s, a, o) = \min_\sigma \mathop{\mathrm{E}}_{\pi,\sigma} \left\{ \sum_{i=0}^{+\infty} \gamma^i r_{t+i} \middle| X_t = s, a, o \right\}. \tag{3}$$

V_π and Q_π are linked by the **Bellman equations for Markov games** [6, 1, 10]

$$V_\pi(s) = \min_{o \in O(s)} \mathrm{SQ}_\pi(s, o) \tag{4}$$

$$Q_\pi(s, a, o) = r_s(a, o) + \gamma \sum_{s' \in S} T_{s,a,o,s'} \min_{o' \in O(s')} \mathrm{SQ}_\pi(s', o') \tag{5}$$

where $\mathrm{SQ}_\pi(s, o) = \sum_{a \in A(s)} \pi_s(a) Q_\pi(s, a, o)$. Notice that $r_s(a, o)$ is outside the sum for Q_π: we only need the reward received upon executing the actions for a stochastic approximation of the Q **values**.

In this work, we propose a new algorithm and compare it with existing ones. These algorithms are based on the second Bellman equation Eq. 5: they only

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

differ by their hypotheses and methods to compute the optimal policy π^*. They all use the general template ALG. 1 where the learning consists of **epochs**: $\mathcal{A}g$ perceives s, chooses a, observes $\mathcal{O}pp$'s action o, receives a reward r, perceives the next state s' and then learns. Notice that only the implementation of the learning stage is shown in the algorithms.

Input: A RL problem $\langle \mathcal{E}, r, R \rangle$
Output: $\widehat{V^*}$,$\widehat{\pi^*}$
1. Initialise V or/and miscellaneous data structures
2. $s \leftarrow$ current state of $\mathcal{E}$
3. **repeat**
4. Choose and take action a
5. $o, r, s' \leftarrow$ action taken by $\mathcal{O}pp$, gained reward, resulting state of $\mathcal{E}$
6. Learn from the epoch
7. $s \leftarrow s'$
8. **until** some convergence criterion is satisfied
9. **return** V_π,π

Algorithm 1: Online Markov games algorithms template.

The following three standard algorithms are often used in this context. The one-agent Q**-learning** [5] embeds the opponent into the environment and learns to beat it specifically, but is weak against new opponents. **Minimax** [11] and **minimax-Q** [6] exploit the links between RL and game theory to express the RL problem as a set of matrix games, one for each state $s \in \mathcal{S}$. The former's pure policy selects the best action in a maximin sense and is therefore sub-optimal when a mixed policy is required. The latter uses LP and achieves optimality, but requires calls to a LP solver.

3 $\mathcal{QL}_2$: a Constrained Optimisation Approach

We now introduce the contribution of this paper: $\mathcal{QL}_2$. This algorithm formulates the RL problem as a constrained optimisation problem, the constraints being the Bellman equations EQ. 4 and EQ. 5, and optimises

$$V_\pi(s_{init}) = \min_{\sigma} \mathop{\mathrm{E}}_{\pi,\sigma} \left\{ R_0 \middle| X_0 = s_{init} \right\}. \tag{6}$$

If we introduce the change of variables $\pi_s^\theta(a) = e^{\theta_s(a)} / \sum_{b \in A(s)} e^{\theta_s(b)}$, where $\theta_s(a) \in]-\infty, +\infty[$ such that the bigger $\theta_s(a)$, the bigger $\pi_s^\theta(a)$ (and *vice versa*), and choose to optimise these values, we obtain the Lagrangian $\mathcal{L}_{\pi^\theta}$ associated to the RL problem which is non-linear in term of the θ values

$$\min_{o \in O(s_{init})} \mathrm{SQ}_{\pi^\theta}(s_{init}, o) + \sum_{s \in \mathcal{D}} \sum_{a \in A(s)} \sum_{o \in O(s)} \lambda_{s,a,o} Q_{\pi^\theta}(s,a,o) + \sum_{s \in \mathcal{S} \setminus \mathcal{D}} \sum_{a \in A(s)} \sum_{o \in O(s)}$$

$$\lambda_{s,a,o} \left[Q_{\pi^\theta}(s,a,o) - r_s(a,o) - \gamma \sum_{s' \in \mathcal{S}} \left[T_{s,a,o,s'} \min_{o' \in O(s')} \mathrm{SQ}_{\pi^\theta}(s', o') \right] \right]. \tag{7}$$

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

where $\mathcal{D}$ is the set of absorbing states. Using a gradient-ascent scheme [12], the $\mathcal{QL}_2$ algorithm presented in ALG. 2 optimises $\mathcal{L}_{\pi^\theta}$ with a surprisingly simple and cheap update rule. The $\alpha_{\theta_s(a)}$ values are updated according to the rule

$$\alpha_{\theta_s(a)} \leftarrow \begin{cases} \alpha_{\theta_s(a)} * inc & \text{if } \Delta\theta_s(a)_t * \Delta\theta_s(a)_{t-1} > 0 \\ \alpha_{\theta_s(a)} / dec & \text{if } \Delta\theta_s(a)_t * \Delta\theta_s(a)_{t-1} < 0 \end{cases} \tag{8}$$

where $inc, dec > 1$. Note that in practice, we impose $|\theta_s(a)| \leq \theta_{max}$.

1. $Q_\pi(s,a,o) \leftarrow (1-\alpha)Q_\pi(s,a,o) + \alpha\left[r + \gamma \min_{o' \in O(s')} \mathrm{SQ}_\pi(s',o')\right]$
2. **for all** $a \in A(s)$ **do**
3. $\quad M(s) \leftarrow \left\{ o \in O(s) \middle| \mathrm{SQ}_{\pi^\theta}(s,o) = \min_{o' \in O(s)} \mathrm{SQ}_{\pi^\theta}(s,o') \right\}$
4. $\quad \theta_s(a) \leftarrow \theta_s(a) - \alpha_{\theta_{s,a}} \frac{\pi^\theta_s(a)}{|M(s)|} \sum_{o \in M(s)} [\mathrm{SQ}_{\pi^\theta}(s,o) - Q_{\pi^\theta}(s,a,o)]$
5. $\quad$ Update $\alpha_{\theta_{s,a}}$
6. **end for**

Algorithm 2: Learning part of the $\mathcal{QL}_2$ algorithm.

4 Experiments

We will now assess the efficiency of Q-learning, Minimax, minimax-Q and $\mathcal{QL}_2$ on $\mathcal{SC}$, a simplified soccer game [6, 10] made up of a 5×4 grid. The reward is either +1 for win, 0 for a draw and −1 for a loss; 0 otherwise. The moves are simultaneous and there can be only one player by cell. When starting the game, the players are placed at random in the first and last columns, and the ball owner is randomly chosen (see FIG. 1(a)). If both players try to move toward the same cell, the one which actually moves is chosen at random (and remains or becomes the ball owner). $\mathcal{SC}$ is therefore a stochastic game. In order to avoid deadlocks, the game has a probability 10^{-2} to end at each step [6].

Every optimal policy for $\mathcal{SC}$ must be mixed. Indeed, in the situation of FIG. 1(b), if π prescribes to go up or to go down, there is a possibility that $\mathcal{O}pp$ anticipates it and does the same action, barring $\mathcal{A}g$'s way. On the contrary, if π prescribes equiprobably these two actions, any prediction by $\mathcal{O}pp$ becomes impossible and $\mathcal{A}g$ will inevitably create an opening.

4.1 Test Protocol

Our test protocol is inspired by [6, 10]. First, an agent using minimax-Q, called mQ-ref, is trained against a random policy $\mathcal{R}nd$ for 10^6 epochs to be used as reference for performance evaluation. Then two instances of each algorithm (including minimax-Q) learn by playing respectively against (i) $\mathcal{R}nd$ and (ii) an agent which is simultaneously learning with the same algorithm. This phase also takes 10^6 epochs, but every 5×10^4 epochs, each agent is tested in 1000 one-to-one contests against mQ-ref to estimate $R_{\text{mQ-ref}}$, i.e. the return that $\mathcal{A}g$ can *really*

ESANN'2008 proceedings, European Symposium on
Artificial Neural Networks - Advances in Computational Intelligence and Learning
Bruges (Belgium), 23-25 April 2008, d-side publi., ISBN 2-930307-08-0.

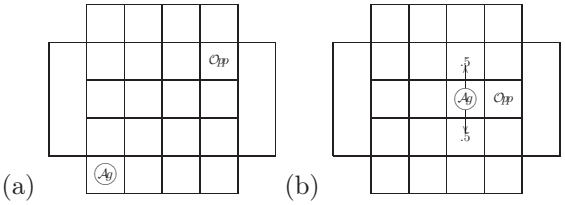

Figure 1: Example of (a) starting set and (b) mixed policy for a 5×4 soccer game.

expect in s_{init} against mQ-ref. This experiment was run 10 times to estimate the mean (the 95% confidence intervals remain small but are not plotted).

The actions are chosen using the **ϵ-greedy** scheme [1] with $\epsilon = .9$ to enhance exploration and value propagation. $\mathcal{QL}_2$'s parameters are $dec = 1.1$, $inc = 1.01$ and $\theta_{max} = 3$. The α values follow a geometric progression and decrease by 10^3 after 10^6 visits.

4.2 Results on $\mathcal{SC}$ with a Random Opponent for the Learning Stage

FIG. 2(a) shows $R_{\text{mQ-ref}}$ for Q-learning, Minimax, minimax-Q and $\mathcal{QL}_2$ learning against $\mathcal{Rnd}$. Q-learning is significantly worse than mQ-ref ($R_{\text{mQ-ref}} \approx -.2$) because it assumes that any opponent acts as $\mathcal{Rnd}$. Minimax is better ($R_{\text{mQ-ref}} \approx -.08$): it relies on the rational player hypothesis and is more careful. However, it lacks a mixed policy: minimax-Q and $\mathcal{QL}_2$ both achieve optimality ($R_{\text{mQ-ref}} \approx 0$) with similar convergence speed.

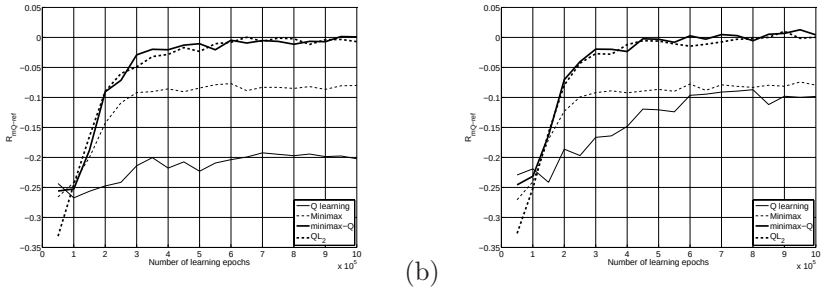

Figure 2: $R_{\text{mQ-ref}}$ for Q-learning, Minimax, minimax-Q and $\mathcal{QL}_2$ when the learning stage opponent (a) is $\mathcal{Rnd}$ or (b) uses the same algorithm.

4.3 Results on $\mathcal{SC}$ with a Learning Opponent for the Learning Stage

FIG. 2(b) shows $R_{\text{mQ-ref}}$ for Q-learning, Minimax, minimax-Q and $\mathcal{QL}_2$ learning against an agent which is simultaneously learning with the same algorithm. Q-learning is still sub-optimal but has improved ($R_{\text{mQ-ref}} \approx -.1$) thanks to its learning stage opponent which in fact determines its strength. On the contrary,

Minimax, minimax-Q and $\mathcal{QL}_2$'s convergence has only slightly accelerated (this appears more obviously on graphs of wins percentages not shown here): they obtain policies of identical strength whatever the opponent since they consider the rational player hypothesis. In particular, Minimax remain sub-optimal.

5 Conclusion

We have shown that Q-learning produces sub-optimal opponent-dependent policies; it however improves greatly when it learns against a learning opponent ([6] made the same observation). As for two-Agent algorithms, we have shown that pure policies (e.g. Minimax) may be sub-optimal as predicted by game theory. minimax-Q and $\mathcal{QL}_2$ both achieve optimality at comparable speed, but $\mathcal{QL}_2$'s updating rule is much more simple.

$\mathcal{QL}_2$ can be improved: it is not yet robust enough and can fail to converge if θ_{max} is too large or too small, due to machine precision limits and entropy constraints on the policy. The impact of θ_{max} remains to be better understood and the gradient-ascent scheme itself should be improved. Moreover, it would be interesting to investigate the suitability of our approach for n-players Markov games where cooperation is conceivable [13] or $n > 2$ [14].

References

[1] R. S. Sutton and A. G. Barto. *Reinforcement Learning: an Introduction*. MIT Press, 1998.

[2] T. M. Mitchell. *Machine Learning*. McGraw-Hill International Editions, 1997.

[3] D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[4] L. P. Kaelbling, M. L. Littman, and A. P. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[5] C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[6] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning (ICML-94)*, pages 157–163, 1994.

[7] P. D. Straffin. *Game Theory and Strategy*. Mathematical Association of America, 1996.

[8] `Qash`: a `Python` generic framework for reinforcement learning in n-player Markov games. http://www.ucl.ac.be/mlg/index.php?page=Softwares.

[9] H. C. Tijms. *A First Course in Stochastic Models*. John Wiley & Sons, 2003.

[10] M. G. Lagoudakis and R. Parr. Value function approximation in zero-sum markov games. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI-2002)*, pages 283–292, 2002.

[11] S. Russel and P. Norvig. *Artificial Intelligence: a Modern Approach*. Prentice Hall, 2003.

[12] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.

[13] M. L. Littman. Friend-or-foe Q-learning in general-sum games. In *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*, pages 322–328, 2001.

[14] M. Tan. Multi-agent reinforcement learning: Independent vs. cooperative learning. In *Readings in Agents*, pages 487–494. Morgan Kaufmann, 1997.