Multi-agent Learning and the Reinforcement Gradient

Michael Kaisers and Karl Tuyls

Maastricht University, P.O. Box, 6200 MD Maastricht,
Michael.Kaisers@maastrichtuniversity.nl,
WWW home page: http://www.michaelkaisers.com

Abstract. The number of proposed reinforcement learning algorithms appears to be ever-growing. This article tackles the diversification by showing a persistent principle in several independent reinforcement learning algorithms that have been applied to multi-agent settings. While their learning structure may look very diverse, algorithms such as Gradient Ascent, Cross learning, variations of Q-learning and Regret minimization all follow the same basic pattern. Variations of Gradient Ascent can be described by the projection dynamics and the other algorithms follow the replicator dynamics. In combination with some modulations of the learning rate and deviations for the sake of exploration, they are primarily different implementations of learning in the direction of the reinforcement gradient.

Keywords: Reinforcement Learning, Evolutionary Game Theory, Dynamical Systems, Gradient Learning

1 Introduction

Looking at the publications of major conferences in the field of multi-agent learning, the number of proposed multi-agent learning algorithms is constantly growing. Many domain-specific problems are circumvented by modifying the learning algorithms in question for the task at hand. A survey of well-established multi-agent learning algorithms with their various purposes is give in [5], and demonstrates the need for a comprehensive understanding of their similarities and differences. In order to compliment empirical comparisons between multi-agent learning algorithms [15], this article will decompose several learning algorithms structurally and deliver qualitative insights. The diversity of learning algorithms makes it imperative to specify the assumptions and *learning bias* which precede any discussion [6].

This article considers independent reinforcement learning algorithms (or learning with *Minimal Information*) applied to multi-agent games, and points out the prevailing principle of *learning along the reinforcement gradient* within a seemingly diverse set of algorithms. In particular, Cross Learning [3, 7], Regret Minimization [9], as well as variations of Gradient-Ascent [1, 4, 12] and Q-learning [8, 14] are considered. In order to demonstrate their inherent similarity, each algorithm is linked to a dynamical system by taking an infinitesimal learning rate. These links have been established by previous work, but the origin of their similarity has not been discussed satisfactorily.

Q-learning has been linked to a dynamical system which allows decomposing of the learning dynamics into exploitation and exploration terms [14]. This reveals that the exploitation terms, which move the behavior toward higher payoff, are equivalent to Cross Learning as described in [3]. The remaining terms allow for inferior strategies to be explored. This article will extend this decomposition, and further relate Cross Learning to the most fundamental concept of learning to increase payoff – learning along the reinforcement gradient.

The remainder of this article is structured as follows: Section 2 formally introduces the concepts of Game Theory that are used in the analysis. The reinforcement gradient is introduced and linked to Gradient-Ascent in Section 3. Subsequently, Section 4 relates the gradient to the replicator dynamics, which is the basis of several learning algorithms. Finally, Section 5 concludes the article.

2 Background

Game theory models strategic interactions in the form of games. Each player has a set of actions, and a preference over the joint action space which is captured in the numerical payoff signal. For two-player games, the payoffs can be given in a bi-matrix (A, B), that gives the payoff for the row player in A, and the column player in B (see Figure 1). In this example, the row player chooses one of the two rows, and the column player chooses either column, and the outcome of this joint action determines the payoff to both.

Players repeatedly interact and receive a payoff according to their joint action. At any time t, the policy $\pi^p = (\pi_1^p, \pi_2^p, \ldots, \pi_i^p, \ldots)$ of a player p denotes the probability distribution over the available actions, where π_i^p represents the probability of choosing action i. The joint policy $\pi = (\pi^1, \pi^2, \ldots)$ describes the behavior of all players at a specific time instant. For ease of notation, we may refer to the joint policy of all opponents of player p as π^{-p}. The expected reward that player p gets, when all players follow a joint policy π is denoted as $V^p(\pi^p, \pi^{-p})$.

3 The Reinforcement Gradient

The reinforcement gradient is used in Gradient-Ascent. Let α be the learning rate. Player p updates its policy π^p at time t according to the increase of the

$$\begin{pmatrix} A_{11}, B_{11} & A_{12}, B_{12} \\ A_{21}, B_{21} & A_{22}, B_{22} \end{pmatrix}$$

Fig. 1: General payoff bi-matrix (A, B) for two-agent two-action games.

expected reward $V^p(\pi^p, \pi^{-p})$ that a change in π^p induces against opponents following their joint policy π^{-p}.

$$\pi^p(t+1) \leftarrow \pi^p(t) + \alpha \frac{\partial V^p(\pi^p, \pi^{-p})}{\partial \pi^p}$$

This implies for any action i:

$$\pi_i^p(t+1) \leftarrow \pi_i^p(t) + \alpha \frac{\partial V^p(\pi^p, \pi^{-p})}{\partial \pi_i^p}$$

The i^{th} element of the gradient can be calculated as the partial derivative of V with respect to π_i. Let e_i denote the i^{th} unit vector. Using the regular differential would lead out of the policy space, hence the differential δe_i first needs to be projected onto the policy space using the orthogonal projection function $\Phi(\zeta) = \zeta - \frac{1}{n}\sum_j^n \zeta_j$, where n is the number of actions.

$$\begin{aligned}\frac{\partial V(\pi, \sigma)}{\partial \pi_i} &= \lim_{\delta \to 0} \frac{[\pi + \Phi(\delta e_i)]\, A\sigma - \pi A\sigma}{\delta} \\ &= \Phi(e_i) A\sigma \\ &= e_i A\sigma - \frac{1}{n}\sum_j^n e_j A\sigma\end{aligned}$$

These dynamics are also known as the *Projection Dynamics* [10]. They are very similar to the replicator dynamics, which are discussed in the context of several learning algorithms within the next section.

4 Algorithms and the Reinforcement Gradient

This section first presents three variations of gradient learning, which involve the gradient in their very definition. Subsequently, three other reinforcement learning algorithms are discussed: Cross Learning as an example policy learner, and Frequency Adjusted Q-learning and Regret Matching as examples of value based learning. These three algorithms yield the replicator dynamics as a core element of their learning behavior. The replicator dynamics are in turn very close to the reinforcement gradient.

Recall the **Gradient-Ascent** update rule presented in the previous section, and let $u = (\frac{1}{n}, \ldots, \frac{1}{n})$.

$$\pi_i^p(t+1) \leftarrow \pi_i^p(t) + \alpha \frac{\partial V^p(\pi^p, \pi^{-p})}{\partial \pi_i^p} = \pi_i^p(t) + \alpha \left[e_i A\sigma - uA\sigma\right]$$

The change $\Delta\pi_i^p(t) = \pi_i^p(t+1) - \pi_i^p(t)$ can be written as:

$$\Delta\pi_i^p(t) = \alpha \left[e_i A\sigma^T - uA\sigma^T\right]$$

The projection dynamics do ensure that $\sum_i \pi_i = 1$, i.e., the policy remains in the tangent space of the policy space. However, it is not guaranteed that $\forall i, 0 \leq \pi_i \leq 1$. Hence, $\Delta\pi_i^p(t)$ may lead the policy out of the valid space of probability distributions, and the actual update needs to be projected onto the closest point in the policy space according to $\pi^p(t+1) = projection(\pi^p + \Delta\pi^p)$. This also holds for the variants of Gradient Ascent unless stated otherwise.

The policy update rule **Infinitesimal Gradient-Ascent** (IGA) [12] is equivalent to Gradient-Ascent with learning rate $\alpha \to 0$. The policy trajectory becomes a continuous dynamical system, where $\dot{\pi}_i^p(t)$ denotes the change in time.[1]

$$\dot{\pi}_i^p(t) = \alpha \underbrace{\left[e_i A \sigma^T - u A \sigma^T\right]}_{\text{gradient learning}}$$

For two-agent matrix games, where $\pi^p = (x, 1-x)$, $\sigma = (y, 1-y)$ and $h = (1, -1)$ the special case can be rewritten as follows:

$$\dot{x} = \alpha(1-x)\left[yhAh^T + A_{12} - A_{22}\right]$$

The modification **Win or Learn Fast** (WoLF) IGA [4] uses exactly the same update rule, with the only modification that α may take two distinct values: If the strategy π^p receives a higher expected reward against opponents playing π^{-p} than an arbitrarily selected Nash Equilibrium strategy π^{p-e}, then $\alpha = \alpha_{min}$. Otherwise, the strategy is doing worse than the Nash equilibrium strategy and $\alpha = \alpha_{max}$, where $\alpha_{min} < \alpha_{max}$. Naturally, this modification merely concerns the speed of learning and not the direction.

$$\dot{\pi}_i^p(t) = \left[e_i A \sigma^T - u A \sigma^T\right] \begin{cases} \alpha_{min} & \text{if } V^p(\pi^p, \pi^{-p}) > V^p(\pi^{p-e}, \pi^{-p}) \\ \alpha_{max} & \text{otherwise} \end{cases}$$

This modulation of the learning rate has been shown to lead to convergence of WoLF IGA players in two-agent matrix games in self-play. However, WoLF IGA requires the estimation of Nash equilibrium strategies and corresponding payoffs, hence it does not comply with the Minimal Information learning bias. The modification **Weighted Policy Learning** [1] introduces a similar modulation of the learning rate but only based on the reinforcement gradient.

$$\dot{\pi}_i^p(t) = \alpha \left[e_i A \sigma^T - u A \sigma^T\right] \begin{cases} \pi_i^p & \text{if } \frac{\partial V^p(\pi^p, \pi^{-p})}{\partial \pi_i^p} < 0 \\ (1 - \pi_i^p) & \text{otherwise} \end{cases}$$

In the two-agent matrix game it takes the following form:

$$\dot{x} = \alpha(1-x)\left[yhAh^T + A_{12} - A_{22}\right] \begin{cases} x & \text{if } \frac{\partial V^p(\pi^p, \sigma)}{\partial x} < 0 \\ (1-x) & \text{otherwise} \end{cases}$$

[1] The limit can be derived in several ways: Either the limit of $\alpha \to 0$ makes α disappear in the resulting equation, or decomposing $\alpha = \zeta\alpha'$ and taking the limit of $\zeta \to 0$ retains a learning rate parameter in the equation.

This modification of IGA actually resembles **Cross Learning** [7], which has been linked to the replicator dynamics [3]. Cross Learning directly updates policy π^p based on the perceived reward $r_i(t)$ it receives after selecting action i.

$$\pi^p(t+1) \leftarrow [1 - \alpha r_i(t)]\, \pi^p(t) + \alpha r_i(t) e_i$$

The learning rule is equivalent to a linear reward-inaction Learning Automata [15], for other variations compare to [9]. The continuous dynamical system is very similar to the one of Gradient Ascent.

$$\dot{\pi}_i^p(t) = \pi_i^p \alpha \underbrace{\left[e_i A \sigma^T - \pi^p A \sigma^T\right]}_{\text{replicator dynamics}}$$

In particular, the two-agent version of Cross Learning takes the following form, which features the product of the two possible learning rate modulations of Weighted Policy Learning:

$$\dot{x} = \alpha x(1-x)\left[yhAh^T + A_{12} - A_{22}\right]$$

It is worth noting here, that the modulation of the learning rate in both Weighted Policy Learning and Cross Learning ensure that the learning update steps become smaller as the policy approaches the boundary of the probability space. As a result, the *project* function is not required.

The relation between policy learning and gradient learning may still be considered rather straight forward. However, also value based learning is based on the same dynamics although the learning update rules appear to be very different. **Q-learning** does not update the policy vector directly. It has been invented for single-agent multi-state environments [16], but will here be discussed in the multi-agent single-state version. The Q-learner repeatedly interacts with its environment, performing action i at time t, and receiving reward $r_i(t)$ in return. It maintains an estimation $Q_i(t)$ of the expected discounted reward for each action i. This estimation is iteratively updated according to the following equation, known as the Q-learning update rule, where α denotes the learning rate and γ is the discount factor:

$$Q_i(t+1) \leftarrow Q_i(t) + \alpha\left(r_i(t) + \gamma \max_j Q_j(t) - Q_i(t)\right)$$

Let k be the number of actions, and let π_i^p denote the probability of selecting action i, such that $\sum_{i=1}^k \pi_i^p = 1$. Furthermore, let $\pi^p(Q) = (\pi_1^p, \ldots, \pi_k^p)$ be a function that associates any set of Q-values with a policy. The most prominent examples of such policy generation schemes are the ϵ-greedy and the Boltzmann exploration scheme [13]. This article exclusively discusses Q-learning with the Boltzmann exploration scheme. Boltzmann exploration is defined by the following function, mapping Q-values to policies, and balancing exploration and exploitation with a temperature parameter τ:

$$\pi_i^p(Q, \tau) = \frac{e^{\tau^{-1} Q_i}}{\sum_j e^{\tau^{-1} Q_j}}$$

The parameter τ lends its interpretation as temperature from the domain of physics. High temperatures lead to stochasticity and random exploration, selecting all actions almost equally likely regardless of their Q-values. In contrast to this, low temperatures lead to high exploitation of the Q-values, selecting the action with the highest Q-value with probability close to one. Intermediate values prefer actions proportionally to their relative competitiveness.

The variation **Frequency-Adjusted Q-learning** (FAQ-learning) [8] modulates the update rule inversely proportional to π_i^p.

$$Q_i(t+1) \leftarrow Q_i(t) + \frac{1}{\pi_i^p}\alpha \left(r_i(t) + \gamma \max_j Q_j(t) - Q_i(t) \right)$$

This update rule leads to a dynamical system that can be decomposed into exploration and exploitation terms, where the temperature parameter τ tunes the balance between the two.

$$\dot{\pi}_i^p = \tau^{-1}\pi_i^p \alpha \underbrace{\left[e_i A\sigma^T - \pi^p A\sigma^T \right]}_{\text{exploitation}} + \alpha\pi_i^p \underbrace{\left(\sum_k \pi_k^p \log \pi_k^p - \log \pi_i^p \right)}_{\text{exploration}}$$

The Polynomial Weights algorithm implements **Regret Minimization** [2, 9]. Player p maintains a set of weights w^p, where $w_i^p(t)$ denotes the loss of playing i rather than the best action in hindsight. These weight are updated after computing loss $l_i(t)$ according to the following equation.

$$w_i^p(t+1) \leftarrow w_i^p(t)\left[1 - \alpha l_i(t)\right]$$

The policy is generated from the weights by normalization.

$$\pi_i^p(w^p) = \frac{w_i^p}{\sum_j w_j^p}$$

Despite the great difference in update rule and policy generation, the infinitesimal limit has been linked to a dynamical system of the following form, which reveals its relation to gradient learning:

$$\dot{\pi}_i^p(t) = \frac{\alpha\pi_i^p \left[e_i A\sigma^T - \pi^p A\sigma^T \right]}{1 - \alpha\left[\max_k e_k A\sigma^T - \pi^p A\sigma\right]}$$

The denominator can be interpreted as a learning rate modulation dependent on the best action's learning update.

All of the algorithms described in this section reveal dynamics that increase their payoff by either following the reinforcement gradient or the replicator dynamics. The two dynamics are closely related. Basically, the replicator dynamics are an asynchronous implementation of the gradient. The terms of the gradient appear to be the foundation of independent reinforcement learning, with learning rate modulations and some deviations for the sake of exploration and coordination.

5 Discussion and Conclusions

The main contributions of this article can be summarized as follows: First, it shows how the concept of learning along the reinforcement gradient persists throughout various independent reinforcement learning algorithms. Second, it demonstrates the qualitative insights that have been facilitated by the established link between the algorithms and their corresponding dynamical system, obtained by taking the limit to an infinitesimal learning rate. Although the algorithms have been discussed in the context of primarily matrix games, their similarity is not limited to this domain but rather inherent to the algorithms.

This article attempts to bring structure into a field of seemingly continuous diversification. The number of proposed learning algorithms is continuously increasing, and we deem recognizing persistent principles such as *learning along the reinforcement gradient* crucial to the integrity of the field. We aim to deepen the understanding of this principle in future work, also substantiating the analysis with empirical support.

Acknowledgements

We thank Daan Bloembergen for the fruitful discussions and useful insights that helped mature the analysis to its current state. This research was partially sponsored by a TopTalent2008 grant of the Netherlands Organisation for Scientific Research (NWO).

References

1. S. Abdallah and V. Lesser. A multiagent reinforcement learning algorithm with non-linear dynamics. *Journal of Artificial Intelligence Research*, 33(1):521–549, 2008.
2. A. Blum and Y. Mansour. *Learning, regret minimization and equilibria.* Cambridge University Press, 2007.
3. T. Börgers and R. Sarin. Learning through reinforcement and replicator dynamics. *Journal of Economic Theory*, 77(1), November 1997.
4. Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136:215–250, 2002.
5. L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172, 2008.
6. J.W. Crandall, A. Ahmed, and M.A. Goodrich. Learning in repeated games with minimal information: The effects of learning bias. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.
7. J.G. Cross. A stochastic learning model of economic behavior. *The Quarterly Journal of Economics*, 87(2):239, 1973.
8. M. Kaisers and K. Tuyls. Frequency adjusted multi-agent Q-learning. In van der Hoek, Kamina, Lespérance, Luck, and Sen, editors, *Proc. of 9th Intl. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 309–315, May, 10-14, 2010.

9. T. Klos, G. van Ahee, and K. Tuyls. Evolutionary dynamics of regret minimization. *Machine Learning and Knowledge Discovery in Databases*, pages 82–96, 2010.
10. W.H. Sandholm. *Population games and evolutionary dynamics*. MIT press Cambridge, MA, 2010.
11. Y. Shoham, R. Powers, and T. Grenager. If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171(7):365–377, 2007.
12. S. Singh, M. Kearns, and Y. Mansour. Nash convergence of gradient dynamics in general-sum games. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 541–548. Citeseer, 2000.
13. R. Sutton and A. Barto. *Reinforcement Learning: An introduction*. MA: MIT Press, Cambridge, 1998.
14. K. Tuyls, P. J. 't Hoen, and B. Vanschoenwinkel. An evolutionary dynamical analysis of multi-agent learning in iterated games. *Autonomous Agents and Multi-Agent Systems*, 12:115–153, 2005.
15. H. van den Herik, D. Hennes, M. Kaisers, K. Tuyls, and K. Verbeeck. Multi-agent learning dynamics: A survey. In Matthias Klusch, Koen Hindriks, Mike Papazoglou, and Leon Sterling, editors, *Cooperative Information Agents XI*, volume 4676 of *Lecture Notes in Computer Science*, pages 36–56. Springer Berlin / Heidelberg, 2007.
16. C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.