Faculty of Mathematics and Computer Science

Multiagent systems course (MAS 2022)

Overview of Multi-agent Reinforcement Learning in Game Playing

Ștefan Buciu

Department of Computer Science, Babeş-Bolyai University
1, M. Kogalniceanu Street, 400084, Cluj-Napoca, Romania

Abstract

Multi-agent Reinforcement Learning (MARL) is a highly researched topic, which has materialized in practical applications with an extraordinary success. Just in the area of Game AI, there are a lot of algorithms that have achieved superhuman performances, such as: AlphaGo, AlphaGo Zero, MuZero, AlphaStar, OpenAI Five, just to name a few. However, from a theoretical standpoint, the MARL algorithms are quite difficult and somewhat weak in literature. Moreover, MARL researchers will have to face difficult challenges in the near future. But, due to the traction to this research subject, advancements will be surely made.

Keywords:
reinforcement learning, rl, multi-agent rl, marl

1. Introduction

Reinforcement Learning (RL) has seen great success in recent years, especially when accompanied with deep neural networks (DNNs), with the emergence of algorithms such as AlphaGo, AlphaZero Go, and MuZero, where the latter is also capable of playing Atari games. Besides board games, it has also been successfully applied in other areas such as robotic control and autonomous driving [5].

Multi-agent RL (MARL) defines a setting where multiple agents are trying to maximize their own long-term result by taking sequential decisions that interact with the environment and each other. MARL comes in different forms, but the most basic one is called *independent* RL (InRL). In this setting, agents have no knowledge of each other, thus treating all interactions as part of their (”localized”) environment [2].

Depending on the types of settings they address, MARL algorithms can be classified as *fully cooperative*, *fully competitive*, or a *mix of the two*. In the cooperative scenario, for example, agents work together to maximize a common long-term return, but in the competitive setting, agents' returns generally sum up to zero. In the latter setting, agents of both cooperative and competitive settings are involved, having general-sum returns. Due to the fact that the objectives

of all agents are not necessarily aligned, the learning goals in MARL are *multi-dimensional*. This brings up some challenges such as: dealing with equilibrium points, efficiency of communication or coordination, and robustness against potential adversarial agents. Additionally, MARL implies concurrency, thus, when agents are trying to improve their own policies, they are facing a *non-stationary* environment. Moreover, MARL has a *combinatorial nature*, seen in the joint action space which increases exponentially with the number of agents, and which could cause scalability issues [5].

The structure of this report is as follows. The next section is focused on a literature review about core MARL concepts. Then, a discussion about the MARL challenges and applications is brought forward. Lastly, the conclusion section gives an overview on the contents of this report.

2. Literature Review

2.1. Markov Games

Markov games (MGs), also known as stochastic games, are a generalization of the Markovian Decision Process (MDP), that captures the intertwinement of numerous agents.

A Markov game can be defined as a tuple $(\mathcal{N}, \mathcal{S}, \{A^i\}_{i\in\mathcal{N}}, \mathcal{P}, \{R^i\}_{i\in\mathcal{N}}, \gamma)$, where:

- $\mathcal{N} = \{1, \dots, N\}$ denotes the set of $N > 1$ agents
- $\mathcal{S}$ denotes the state space observed by all agents
- A^i denotes the action space of agent i
- Let $A := A^1 \times \cdots \times A^N$, then $\mathcal{P} : \mathcal{S} \times A \to \Delta(\mathcal{S})$ denotes the transition probability from any state $s \in \mathcal{S}$ to any state $s' \in \mathcal{S}$ for any joint action $a \in A$
- $R^i : \mathcal{S} \times A \times \mathcal{S} \to \mathbb{R}$ is the reward function that determines the immediate reward received by agent i for a transition from (s, a) to s'
- $\gamma \in [0, 1)$ denotes the discount factor

$$V^i_{\pi^i,\pi^{-i}}(s) := \mathbb{E}\left[\sum_{t\geq 0} \gamma^t R^i(s_t, a_t, s_{t+1}) \middle| a^i_t \sim \pi^i(\cdot|s_t), s_0 = s\right] \tag{1}$$

Given a time t, and state s_t, every agent $i \in \mathcal{N}$ executes an action a^i_t. After agents' moves are done, the system transitions to state s_{t+1}, and they are rewarded by $R^i(s_t, a_t, s_{t+1})$. Each agent tries to maximize its own long-term reward, by finding the policy $\pi^i : \mathcal{S} \to \Delta(A^i)$ such that $a^i_t \sim \pi^i(\cdot|s_t)$. Thus, the value-function $V^i : \mathcal{S} \to \mathbb{R}$ of agent i becomes a function of the joint policy $\pi : \mathcal{S} \to \Delta(A)$ defined as $\pi(a|s) := \prod_{i\in\mathcal{N}} \pi^i(a^i|s)$. In Equation 1, $-i$ denotes the indices of all agents in $\mathcal{N}$ but agent i. As a result, an MG's solution concept differentiates from that of an MDP, because each agent's optimal performance is determined not only by its own policy, but also by the choices of all other participants in the game [5].

2.2. Nash Equilibrium

In game theory, Nash equilibrium (NE) is one of the most fundamental solution notions. It characterizes an equilibrium point π^*, such that, for any agent $i \in \mathcal{N}$, the policy $\pi^{i,*}$ is the *best response* of $\pi^{-i,*}$ [3].

Given a MG $(\mathcal{N}, \mathcal{S}, \{A^i\}_{i\in\mathcal{N}}, \mathcal{P}, \{R^i\}_{i\in\mathcal{N}}, \gamma)$, a Nash equilibrium can be defined as a joint policy $\pi^* = (\pi^{1,*}, \dots, \pi^{\mathcal{N},*})$, such that for any $s \in \mathcal{S}$ and $i \in \mathcal{N}$, $V^i_{\pi^{i,*},\pi^{-i,*}}(s) \geq V^i_{\pi^i,\pi^{-i,*}}(s)$, for any π^i. NE exists for finite-space infinite-horizon discounted MGs as a common MARL learning goal, although it may not be unique in general. If such an equilibrium point exists, most MARL algorithms are designed to converge to it [5].

2.3. MARL Settings

2.3.1. Cooperative Setting

All agents in a completely cooperative setting normally share a single reward function, such as $R^1 = R^2 = \cdots = R^N = R$. In the AI community, this model is known as multi-agent MDPs (MMDPs), while in the control/game theory field, it is known as Markov teams/team Markov games. Furthermore, this cooperative setting can be understood as a particular example of Markov *potential* games, with the potential function being the common accumulated reward, from a game-theoretic perspective. With this model in mind, all agents' value function and Q-function are similar, allowing single-agent RL algorithms, such as Q-learning update, to be used if all agents work together as a single decision maker. The game's global optimum for cooperation has now reached a Nash equilibrium.

Aside from the common-reward model, team-average reward is a slightly more general and surging paradigm for cooperative MARL. Agents can have distinct reward functions, which can be kept secret to each agent, and the purpose of cooperation is to optimize the long-term reward corresponding to the average reward $\bar{R}(s, a, s') := N^{-1} \cdot \sum_{i \in \mathcal{N}} R^i(s, a, s')$ for any $(s, a, s') \in \mathcal{S} \times A \times \mathcal{S}$. The model above is a special instance of the average reward model, which allows for more heterogeneity across actors. It also protects agent privacy and makes the creation of decentralized MARL algorithms easier. Because of this heterogeneity, communication protocols must be incorporated into MARL, as well as the study of communication-efficient MARL algorithms [5].

2.3.2. Competitive Setting

In MARL, fully competitive settings are often treated as zero-sum Markov games, i.e. $\sum_{i \in \mathcal{N}} R^i(s, a, s') = 0$ for any (s, a, s'). Most of the literature focuses on two competing agents for ease of algorithm analysis and computational tractability, where the reward of one agent is obviously the loss of the other. In addition to direct game-playing applications, zero-sum games can also be used as a model for robust learning, because the uncertainty that obstructs the agent's learning process can be accounted for as a fictitious opponent in the game who is always against the agent. As a result, the Nash equilibrium produces a dependable policy that maximizes the worst-case long-term return.

2.3.3. Mixed Setting

The mixed setting, also known as the general-sum game setting, is one in which the purpose and relationships among agents are not restricted. Each agent is self-interested, and his or her reward may conflict with that of others. The most significant effect on algorithms developed for this general setting comes from game theory equilibrium solution notions like Nash equilibrium. The mixed setting is also considering settings to comprise situations with both completely cooperative and competitive agents, such as two zero-sum competitive teams with cooperative agents in each team.

3. Discussion

3.1. MARL Challanges

Despite having a general model with many applications, MARL has additional theoretical issues in addition to those that arise in single-agent RL. The critical issues that halt the creation of MARL theories are summarized below [5].

3.1.1. Non-Unique Learning Goals

Unlike single-agent RL, where the agent's purpose is to maximize the long-term return as effectively as possible, MARL's learning goals can be a bit unclear at times. In fact, many early MARL efforts had a fundamental flaw: the *unclarity* of the problems being addressed. Indeed, the objectives that must be addressed when analyzing MARL algorithms might be multi-faceted. Convergence to Nash equilibrium is the most prevalent goal. By definition, NE is the position from which no agent will diverge if all algorithms eventually converge. Under the assumption that all agents are rational and capable of flawless reasoning and limitless mutual modeling, this is surely a reasonable solution concept in game theory. With *bounded rationality*, however, the agents may be limited to finite mutual modeling. As a

result, for practical MARL agents, the learning dynamics designed to converge to NE may not be acceptable. Instead, the goal could be to create the optimum *learning strategy* for a certain agent and a set of other agents in the game.

Furthermore, it is debatable whether *convergence* (to the equilibrium point) is the most important performance criterion in MARL algorithm analysis. Indeed, it has long been known that value-based MARL algorithms fail to converge to the *stationary* NE of general-sum Markov games, which led to the development of a new solution concept known as *cyclic equilibrium*, in which the agents cycle rigidly through a set of stationary policies, rather than converging to any NE policy. Alternatively, the learning goal can be split into two parts: *stable* and *rational*, with the former requiring convergence to a best-response when the other agents remain stationary and the latter requiring convergence to a best-response when the other agents remain stationary. Convergence to NE occurs naturally in this situation if all agents are both stable and rational. Furthermore, the concept of *regret* adds a new dimension to capturing agents' rationality by comparing the algorithm's performance to the best hindsight static strategy. The convergence to the equilibrium of particular games is guaranteed by no-regret algorithms with asymptotically zero average regret, which in turn ensures that the agent is not *exploited* by others.

In addition to the goals concerning optimizing the return, several other goals that are special to multi-agent systems have also drawn increasing attention. For example, Foerster et al. [1] investigate learning to communicate, in order for the agents to better coordinate. Such a concern on communication protocols has naturally inspired the recent studies on communication-efficient MARL. Other important goals include how to learn without over-fitting certain agents, and how to learn robustly with either malicious/adversarial or failed/dysfunctional learning agents. Still in their infancy, some works concerning aforementioned goals provide only empirical results, leaving plenty of room for theoretical studies [5].

3.1.2. Non-Stationarity

Another major problem of MARL is that numerous agents frequently learn at the same time, making the environment in which each agent operates non-stationary. The reward of other opponent agents, as well as the evolution of the state, are all affected by one agent's behavior. As a result, the learning agent must account for the actions of the other agents and adjust to their combined behavior accordingly. The stationarity assumption for establishing the convergence of single-agent RL algorithms, specifically, the stationary Markovian property of the environment such that the individual reward and current state depend only on the prior state and action taken, is invalidated as a result.

Indeed, the algorithms may fail to converge if the agent ignores this issue and optimizes its own policy assuming a stagnant environment, which is commonly referred to as an independent learner. Independent learning, on the other hand, has been shown to produce satisfactory results in practice. Non-stationarity has long been regarded in the literature as the most well-known issue in MARL. A recent comprehensive analysis, in particular, provides an overview of how state-of-the-art multi-agent learning algorithms approach and solve it.

3.1.3. Scalability

To deal with non-stationarity, each agent may need to account for the *joint action space*, which grows exponentially in size as the number of agents grows. The *combinatorial nature* of MARL is another term for this. Theoretical analysis of MARL is complicated by the enormous number of agents, particularly the convergence analysis. The fact that MARL theories for the two-player zero-sum situation are far more broad and evolved than those for general-sum settings with more than two agents supports this claim. In terms of action dependence, one possible solution to the scalability problem is to assume the *factorized* structures of either the value or reward functions. Relevant theoretical analysis that incorporates a special dependence structure and builds a probably convergent model-based method had not been created until recently.

The development of theories for deep multi-agent RL is another MARL theoretical challenge that arises independently of, but is amplified by, the scalability issue. Scalability difficulties, in particular, necessitate the use of function approximation, particularly deep neural networks, in MARL. With our current understanding of deep learning theory, let alone deep RL theory, the theoretical study of deep MARL is practically new area, despite its empirical success.

3.1.4. Various Information Structures

The information structure of MARL, meaning, who knows what at the training and execution, is more involved than in the single-agent scenario. In the context of Markov games, for example, observing the instantaneous state

s_t is sufficient for each agent to make judgments because the local policy π^i mapping from $\mathcal{S}$ to $\Delta(A^i)$ contains the equilibrium policy. In contrast, under the typical perfect recall assumption, each actor in an extensive-form game may be required to recall the history of previous judgments. Furthermore, as self-interested agents, each agent has limited access to the opponents' *policies* or rewards, and can only see the action samples they have taken. This incomplete knowledge worsens the problems produced by non-stationarity because the samples are unable to recover the exact behavior of the opponents' underlying policies, increasing the non-stationarity perceived by individual agents. The aforementioned independent learning scheme, which presupposes just the observability of local action and reward and suffers from non-convergence in general, is an extreme case.

3.2. AlphaStar

StarCraft is a real-time strategy game (RTS) where players have to balance high-level economic decisions with individual control of hundreds of units. Many agents have tried to achieve superhuman skill levels, however none but AlphaStar succeeded. The predecessor algorithms were simplifying important aspects of the game, and also giving themselves some advantages such as: utilizing superhuman capabilities, and using hand-crafted sub-systems, but they were not managing to reach the level of top StarCraft players.

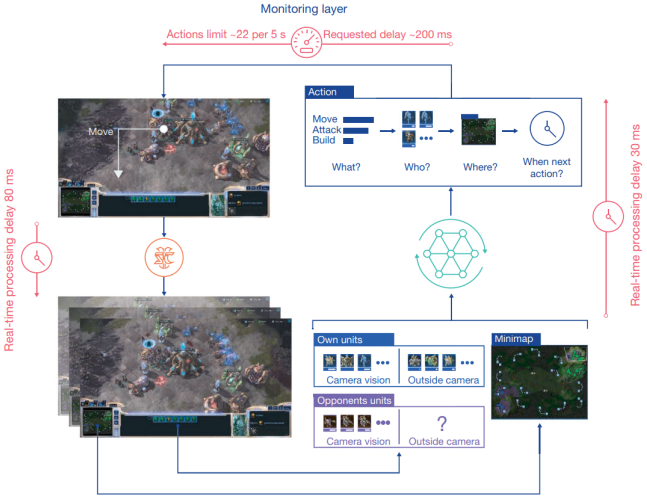

Fig. 1. AlphaStar game observation [4]

AlphaStar is a multi-agent reinforcement learning algorithm which achieved the Grand master level, and ranking above 99.8% of the officially ranked human players. It managed to attain this level a performance by using multiple general-purpose learning methods, a MARL algorithm which uses data from professional human play and from self play, and numerous strategies and counter-strategies which were represented by DNNs [4].

Figure 1 presents the monitoring layer of AlphaStar. At each step t, the system makes an o_t observation which represents a list of all observable units and their attributes. Each action a_t is represented by an action type (e.g., move), a target (e.g., which unit to move), a target location (e.g., where in the camera view to move the unit), and when to observe and act next. Given this action representation, a total number of 10^26 is available at every step. Moreover, there is a special action represented by the moving of the camera view, with the purpose of gathering more information.

The Figure 1 also shows an action limit and a delay. These limitations are put into place to simulate the human physical constraints, which renders down their rate of actions and reaction time.

Due to the high complexity of the StarCraft game, AlphaStar uses multiple new and existing general-purpose techniques for NN architectures, imitation learning, RL, and MA learning. However, a central role in AlphaStar is played by a policy $\pi_\theta(a_t|s_t, z) = \mathbb{P}[a_t|s_t, z]$, which is represented by a NN with parameters θ that receives all observations $s_t = (o_{1:t}, a_{1:t-})$ as inputs and produces actions as output. Additionally, this policy is conditioned by a statistic z which

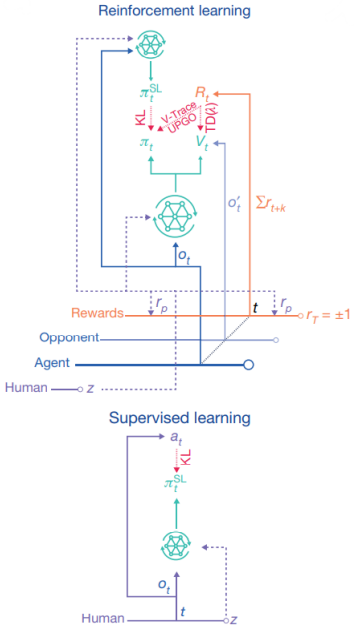

Fig. 2. AlphaStar training [4]

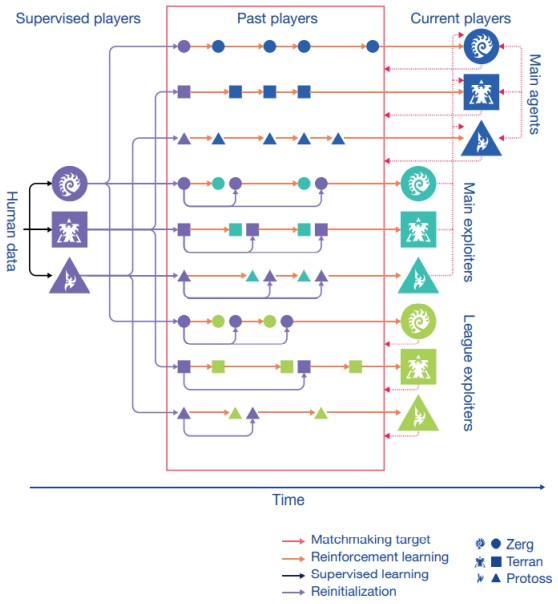

Fig. 3. AlphaStar agent pool [4]

summarizes a human data sampled strategy. To handle StarCraft's complexity, a couple of decisions have been made when building the agent's architecture:

- A self-attention mechanism is used to process observations of player and opponent units.
- Scatter connections are used to integrate spatial and non-spatial information.
- A DLSTM system is used to process temporal sequences of information, to deal with the game's partial observability.
- The agent employs an auto-regressive strategy and a recurrent pointer network to govern the structured, combinatorial action space.

The reinforcement learning technique used by AlphaStar is based on a policy gradient approach that is similar to advantage actor–critic. On repeated experiences, updates were applied asynchronously. This necessitates a strategy known as off-policy learning, which entails modifying the current policy based on earlier policy experience. The observation that present and past policies are exceedingly unlikely to align over many steps in vast action spaces motivates our strategy. We utilize a variety of strategies to learn successfully despite the mismatch: temporal difference learning ($TD(\lambda)$), clipped importance sampling (V-trace), and a new self-imitation algorithm (UPGO) that steers the policy toward better-than-average reward trajectories. The value function is evaluated utilizing information from both the player's and opponent's perspectives during training only to reduce variation.

Usually, in Game AI, the exploitation-exploration dilemma always arises. In the case of StarCraft, focusing on exploitation may be risky since discovering new strategies is challenging. Consider a policy that has mastered the build and use of ground unit micro-tactics. Any deviation that builds and uses air units in an erroneous manner will degrade performance. It's exceedingly unlikely that naive exploration will follow a precise set of instructions that builds air units and effectively employs their micro-tactics over thousands of steps. To address this issue, human data is used. Thus, every agent's parameters are initialized to the ones of the supervised learning agent. Next, during reinforcement learning, agents are conditioned on a statistic z, or they are trained unconditionally (as shown in Figure 2). When conditioned, agents will be rewarded for following the strategy corresponding to z, and will be penalized whenever their action probabilities differ from the supervised policy.

In Figure 3, an MARL algorithm called league training is introduced as an answer to these game-theoretic challenges. Self-play algorithms, like those employed in chess and Go, learn quickly yet can cycle forever without progressing. In two-player zero-sum games, fictitious self-play (FSP) avoids cycles by computing the best response against a uniform mixture of all prior policies; the mixture converges to a Nash equilibrium. This method is extended to find the best response against a non-uniform mixture of opponents. This league of prospective adversaries comprises a wide spectrum of agents, as well as their policies from prior versions. Each iteration involves each agent playing games against opponents drawn from a combination policy tailored to that agent. The actor–critic reinforcement learning process is used to adjust the agent's parameters based on the outcomes of those games.

3.3. Counterfactual Regret Minimization

Counterfactual Regret Minimization (CFR) is by far the most successful family of algorithms for solving complex extensive-form games, as well as a powerful weapon in multi-agent RL's armory. CFR is based on a regret matching algorithm that uses self-play to adaptively learn a coupled equilibrium. Regret matching is an algorithm that analyzes an agent's history and updates their policy in a "what-if" scenario. When an agent chooses rock and loses to paper in a repeated rock-paper-scissor game, for example, it will ponder what would have happened if it had selected paper or scissor instead, and cope with the opponent who likes paper. Choosing the optimal reaction, on the other hand, is problematic: not only can this be exploited by a cunning adversary, but it is also impractical when the calculation of the best response is useless, such as when it takes exponential time. As a result, regret matching produces a stochastic response in which the likelihood of actions is proportional to their cumulative positive regrets, i.e. the payoff gain if the action had been played previously. In the scenario above, the agent will play scissor twice as often as paper in the next round, whereas the actions are picked arbitrarily if no positive regret exists. When the game is played enough times, the distribution of the agents' responses is shown to converge to the correlated equilibrium.

Regret matching can be thought of as an online learning algorithm in which regrets are learned to model the tactics of other agents, and the cumulative regret bound is crucial to its performance and convergence. It has been established that cumulative regret rises in a non-linear relationship with the number of rounds played. In a generalized form of regret matching, the bound of regret is also given, where the probability distribution of actions in the following round can be proportional to the polynomial or exponential form of cumulative regret. [3]

4. Conclusion

Given the prevalence of sequential decision-making with several agents coupled in their actions and knowledge, Multi-Agent Reinforcement Learning (MARL) has long been an active and significant research field in reinforcement learning. Theoretical comprehension of MARL algorithms is well recognized to be difficult and somewhat weak in the literature, in stark contrast to its enormous practical success. Indeed, developing a comprehensive theory for MARL necessitates the use of tools from dynamic programming, game theory, optimization theory, and statistics, all of which are difficult to unite and study in a single context.

References

[1] Foerster, J., Assael, I.A., De Freitas, N., Whiteson, S., 2016. Learning to communicate with deep multi-agent reinforcement learning. Advances in neural information processing systems 29.

[2] Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., Graepel, T., 2017. A unified game-theoretic approach to multiagent reinforcement learning. Advances in neural information processing systems 30.

[3] Lu, Y., Yan, K., 2020. Algorithms in multi-agent systems: A holistic perspective from reinforcement learning and game theory. arXiv preprint arXiv:2001.06487 .

[4] Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., et al., 2019. Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature 575, 350–354.

[5] Zhang, K., Yang, Z., Başar, T., 2021. Multi-agent reinforcement learning: A selective overview of theories and algorithms. Handbook of Reinforcement Learning and Control , 321–384.