Literature Review on Classifications of Multi-Agent Bandits and Multi-Agent Reinforcement Learning

Parajuli, Samyak
samyak.parajuli@berkeley.edu

Tanwisuth, Sandy
kst@berkeley.edu

May 2021

1 Introduction

This paper gives a broad overview of current literature in Multi-Armed Multi-Agent Bandit settings (MAMAB) and Multi-Agent Reinforcement Learning (MARL). There is a history of cross-pollination between the MAMAB and MARL subfields. Typical literature review in each subfield of multi-agent cover different aspects of basis composing the algorithms, and several features of algorithms such as robustness, computation, time; however, current works have not compare the similarities and differences of MAMAB to MARL. Our work not only covers the ground of features comparison but also aims to bridge the theory in bandits with the state-of-the-art practices in deep reinforcement learning. We also covers classifications of behavioral policies, demonstrating the similarities and differences in MAMAB and MARL. We hope that our work initiates meaningful conversations between MAMAB and MARL by put forth the importance notions MARL can learn from MAMAB and vice versa.

2 Formalism

2.1 General Setting and Objectives

We first express the general setting and objectives of the multi-armed bandit and reinforcement learning problem in both the single agent and multi agent case.

2.1.1 Bandit Problem

2.1.1.1 Single Agent A single agent multi-armed bandit is defined as [1]:

- a tuple $\langle A, F \rangle$
- A is a set of arms

- $F(a)$ is a stochastic function providing a reward in the range $[0, 1]$

The objective is to minimize the expected cumulative regret [2]:

$$\sum_{t=1}^{T} \mu^{\star} - \mu_{a(t)}$$

2.1.1.2 Multi Agent

In the multi-agent bandit setting, we have [3]:

- tuple $\langle D, A, F \rangle$ Where D is the set of m enumerated agents.
- A is the set of joint arms, which is the Cartesian product of the sets of actions for each agent i for each of the m agents in D.
- $F(a)$ is a stochastic function providing a global reward when a joint arm is pulled.
- G = (V, E) is a graph which models the communication network connecting agents where Each node corresponds to an agent and each edge between a pair of nodes is a communication path between those agents.

At each time step, each agent selects an arm and receives an independent and identically distributed reward associated with the selected arm. Between rounds agents share information with each other in some fashion.

2.1.2 Reinforcement Learning Problem

2.1.2.1 Single-Agent

A Framework for Single-Agent Learning In the single-agent setting, Markov Decision Process (MDP) is a formal framework describing the interactions between the agent and its environment. Detail descriptions of the MDP formal setting are below.

Markov Decision Process (MDP) MDP [4] is a formal framework for Reinforcement Learning (RL). Consider an MDP: $\mathcal{M} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, an agent observes states from the state space $\mathcal{S}$ according to their observation function $o \in \mathcal{O}$ and select actions from their finite action space $\mathcal{A}$. The state updates according to the transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ where the function determines the probability of a transition from any state $s \in \mathcal{S}$ to any state $s' \in \mathcal{S}$ given any possible action $a \in \mathcal{A}$. The agent receives scalar rewards according to their reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$. The agent then attempts to maximize its expected sum of discounted rewards $\mathbb{E}\{\sum_{t=0}^{\infty} \gamma^t r_t\}$ where $\gamma \in [0, 1]$ is a temporal discount factor that encourages the agent to discount future rewards and r_t is the reward received at timestep t.

2.1.2.2 Multi-Agent

A Framework for Multi-Agent Learning Recent developments in multi-agent reinforcement learning focused on sequential decision making in a game theoretical environment called ***Sequential Social Dilemma (SSD)*** [5]. The ways in which *SSD* is different from the typical matrix game in game theory space is that the set-up of *SSD* (1) considers temporally extended series of actions rather than by individual decisions, (2) perceives cooperative behaviors as graded quantity, and (3) assumes that agents must make decisions with only partial information about other agents in the environment [5, 6]. These characteristics suggest that partially observable Markov games might be a suitable model for MARL [4].

Partially observable Markov decision process (POMDP) defines as $\mathcal{M} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$. The agents observe states from the state space $\mathcal{S}$ according to their observation function $\mathcal{O}$ and select sequence of their actions from the action space $\mathcal{A}$. The state updates according to the transition function $\mathcal{T}$ and agents receive scalar rewards according to their reward function $\mathcal{R}$. Agents attempt to maximize their expected sum of discounted rewards $\mathbb{E}\{\sum_{t=0}^{\infty} \gamma^t r_t\}$ where $\gamma \in [0, 1]$ is a temporal discount factor that encourages agents to discount future rewards and r_t is the reward received at timestep t. Markov games encapsulate multiple reward functions, allowing different agents to receive different rewards. Markov games represent agent behavior as policies $\Pi : \mathcal{O} \to \mathcal{A}$ which maps observations to individual actions. This allows Markov games to better represent intermediate stages in their behavioral policies, during which agents may exhibit confused behavior or behavior that combines multiple strategies.

Markov Game Formulation Partially-observable general-sum Markov game [4, 5, 7] describes formularizations of MARL. At each timestep, agents take actions based on their observation of the environment (but not of each others' internal models), and receive individual rewards based on the population that they belong to. As reinforcement learners, their goal is to learn by experience a behavioral policy that minimizes their delay and consequently maximizes their reward.

SSD's has the assumption that agents are "independent" of each other and model each other implicitly as non-stationarity in the environment [5]. This simplification reduces the effectiveness of the optimal policy by obscuring relevant information from the agents' environment model [6], but is nevertheless commonly used to prevent infinitely recursive modelling [5].

Formally, works in MARL typically define an N-player partially observable Markov game $\mathcal{M}$ with a finite state space $\mathcal{S}$ [5, 8, 7]. The observation function $\mathcal{O} : \mathcal{S} \to \mathbb{R}^d$ maps states to d-dimensional observations that obscure other agents' internal states. Each agent has an action space, and represented collectively as the joint action space $\mathcal{A} : \{A_1, \ldots, A_n\}$. Each joint action $\{a_1, \ldots, a_n\} \in \mathcal{A}$ advances the state according to the deterministic transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$. Each agent i receives a reward $\mathcal{R}_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Each agent's reward depends on the initial state, the joint action and its population.

3 Classifications of Behavioral Policies

Frameworks for Classifications Survey papers in multi-agent have attempted to classify different behaviors in multi-agent learning [9, 10, 11, 12, 13, 14]. Within those classifications, we can distinguish them into two-main subcategories as the game-theoretical based and the deep learning based.

Game Theoretical Classifications Shoham et. al. defined the multi-agent framework into five different agendas including: **computational, descriptive, normative, prescriptive cooperative, and prescriptive non-cooperative** [10]; each components described as computing properties of the game iteratively, investigating formal models of learning that agree with people's behavior, focusing on which repeated-game strategies are in equilibrium, decentralizing the control of a system operating in a dynamic environment by focusing on joint policy and resource allocations, and asking how an agent should act to obtain high reward in the repeated game individually respectively.

Deep Learning Classifications Stone raised a concern that these categories are limited since the provided frameworks only consider multi-agent learning from game theoretical perspectives [12] without taking into consideration of recent developments in deep learning. To address this issue, we adopted the classifications from a recent up-to-date works [13, 14]. Hernandez-Leal et. al. adopted the classifications from the insipirations of [9, 11, 12, 15]. They classify behaviors of the multi-agent learning algorithms into four non-mutually exclusive sub-categories including **emergent behaviors, cooperation and competitions, communication, and inference of other agents.** We explain the details including implications and implementations of these categories in the following section.

Since MAMAB and MARL algorithms are more closely related to the second classifications, we adopted Hernandez-Leal et. al. frameworks [13, 14] while incorporating and comparing both algorithms in MAMAB and MARL that fit into each individual classification. For the purpose of this work, we will only highlight the similarities and differences between the algorithms without going into the implementation details.

3.1 Emergent Behaviors

Emergent behaviors defined as those algorithms that aim to analyze and evaluate Deep Reinforcement Learning (DRL) algorithms. Since this category deals directly with deep learning aspect, to our knowledge, there is no existing algorithms in MAMAB that fit into this category. Nonetheless, recent works developed in MARL that based off of either Deep Q-learning (DQN) [16], or Proximal Policy Optimization (PPO) [17] are well-suited in this category.

3.1.1 Deep Q-learning (DQN)

Deep Q-learning is a neural network architecture based off of the Q-Learning algorithm [18]. Agents calculates the quality Q of the combinations of state $\mathcal{S}$ and action $\mathcal{A}$ in other words, the function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ maps state and action in combinations to a scalar reward on real numbers. The Q-learning algorithm described as follows:

Algorithm 1 Q-learning: Learn function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

Require:
States $\mathcal{S} = \{1, \ldots, s\}$
Actions $\mathcal{A} = \{1, \ldots, s\}$, $A : \mathcal{S} \Rightarrow \mathcal{A}$
Reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
Black-box (probabilistic) transition function $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$
Learning rate $\alpha \in [0, 1]$, typically $\alpha = 0.1$
Discounting factor $\gamma \in [0, 1]$
function Q-LEARNING($\mathcal{S}$, A, R, T, α, γ)
    Initialize $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ arbitrarily
    while Q is not converged **do**
        Start in state $s \in \mathcal{S}$
        while s is not terminal **do**
            Calculate π according to Q and policy $\pi(s) \leftarrow \arg\max_a Q(s, a)$
            $a \leftarrow \pi(s)$
            $r \leftarrow R(s, a)$
            $s' \leftarrow T(s, a)$
            $Q(s', a) \leftarrow (1 - \alpha) \cdot Q(s, a) + \alpha \cdot (r + \gamma \cdot \max_{a'} Q(s', a'))$
            $s \leftarrow s'$
    return Q

3.1.2 Proximal Policy Optimization (PPO)

Schulman et. al. developed Proximal Policy Optimization based off of Policy Gradient method [17]. PPO is an on-policy algorithm and can be used for environments with either discrete or continuous action spaces [19].

Algorithm 2 *PPO*

Require:

initial policy parameters θ_0

initial value function parameters ϕ_0

function PPO(θ_0, ϕ_0)

for for $k = 0, 1, 2, \ldots$ do **do** Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment. Compute rewards-to-go $\hat{R}_t$. Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}. Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k| T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_k}(a_t \mid s_t)} A^{\pi_{\theta_k}}(s_t, a_t), g\left(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)\right) \right)$$

typically via stochastic gradient ascent with Adam. Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k| T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.

return Q

3.1.3 Emergent Behaviors in MARL

Works in MARL can be divided into ones which adopted DQN [5, 8, 20, 7] and the others which adopted PPO [21, 22] as the based-line algorithms. Since these algorithms originally aim for single-agent performance, MARL then use these algorithms as baseline for self-play [20, 21]. And through self-play, different behavavioral policies of agents that seem like cooperations and competitions emerge. More information on cooperation and competition discussed in *section 3.3.2*.

3.2 Communications

Communication is the next fundamental aspect of a multi agent problem where the goal is for agents to maximize their combined utility by means of sharing information. This can come in any combination of a local or global communication protocol that is either discrete or continous.

3.2.1 Distributed Bandits

Communication with neighbors improves performance between agents but the amount of improvement experienced will be based on the structure of the communication network.

[23] initially presents the coopUCB1 and coopUCB2 algorithms that use modified upper confidence decision-making heuristic that captures the effect of the additional information an agent receives through communication with other agents as well as the rate of information propagation through the network. The general expression for upperconfidence methods are:

$$Q_i^k(t-1) = \hat{\mu}_i^k(t-1) + C_i^k(t-1)$$

3.2.1.1 CoopUCB1

$$C_i^k(t-1) = \sigma_g \sqrt{\frac{2\gamma}{G(\eta} \cdot \frac{\hat{n}_i^k(t-1) + \epsilon_c^k}{M\hat{n}_i^k(t-1)} \cdot \frac{\ln(t-1)}{\hat{n}_i^k(t-1)}}$$

3.2.1.2 CoopUCB2

$$C_i^k(t-1) = \sigma_g \sqrt{\frac{2\gamma}{G(\eta} \cdot \frac{\hat{n}_i^k(t-1) + f(t-1)}{M\hat{n}_i^k(t-1)} \cdot \frac{\ln(t-1)}{\hat{n}_i^k(t-1)}}$$

The difference between these two is that the UCB heuristic for coopUCB2 depends on the total number of agents M, but not the epsilon term which is a function of the global graph structure.

3.2.1.3 CoopUCL They also derive a Bayesian algorithm coopUCL which is the distributed version of the single agent UCL algorithm. In this setting we have:

$$Q_i^k(t-1) = \hat{\nu}_i(t-1) + \hat{\sigma}_i(t-1)\Phi^{-1}(1-\alpha(t-1))$$

Where at each time t, Q, the upper-credible-limit is computed for each arm. This is an upper bound computed with $\hat{\nu}$ and $\hat{\sigma}$ the posterior mean and standard deviation which holds with probability $\alpha(t-1)$

3.2.2 Loose Couplings

These class of multi agent multi armed bandit algorithms focuses on the fact that the joint action space scales exponentially with the number of agents in the system.

So instead of considering an agent that decides on the actions of all agents involved, we can exploit the fact that many coordination tasks have loose couplings. This means the global reward function can be decomposed into ρ possibly overlapping noisy, observable and independent local reward functions over subsets of agents [24] - $f = \sum_{e=1}^{\rho} f^e$

In this loose coupling setting, the communication network is described as a bipartite graph $G = \langle D, \{f^e\}_{e=1}^{\rho}, E\rangle$, where the nodes are agents D and components of the factored reward function, an edge $\langle i, f^e\rangle$ exists if and only if agent i influences component f^e and $\mu(a) = \sum_{e=1}^{\rho} \mu^e(a^e)$ is the mean of a joint arm.

Conflicts between overlapping groups may arise since the optimal local arms for an agent in two groups may differ. Therefore, these methods need to define the argmax-operator that can deal with the factored representation of a MAMAB, while still returning the full joint arm that maximizes the sum of samples. They do this using variable elimination methods which compute the joint arm that maximizes the global reward without explicitly enumerating over the full joint arm by sequentially eliminates an agent from the communication graph, while computing its best response with respect to its neighbours.

3.2.2.1 Multi-Agent Upper Confidence Exploration (MAUCE) MAUCE executes a joint action at every timestep for a particular factorization to maximize both the estimated mean reward plus an exploration bonus.

The algorithm improves upon the combinatorial bandit framework [25] where arms with unknown distributions form super arms. MAUCE achieves a regret bound that depends on the harmonic mean of the local upper confidence bounds, rather than their sum.

3.2.2.2 Multi-Agent Thompson Sampling (MATS) At each time step t, MATS draws a sample $\mu_t^e(a^e)$ from the posterior for each group and local arm given the history, $\mathcal{H}_{t1}$, of local actions and rewards associated with past pulls. MATS samples the local mean rewards according to the beliefs of the user at each time step, and then uses variable estimation to maximize in order to find the optimal joint arm.

MATS achieves a regret bound that scales sublinearly with a factor AT, where A is the number of local arms.

3.2.3 Communications in MARL

Communications in MARL are usually defined as POMDPs with cooperative agents. Recent works in this subcategory tend to focus on optimal joint policies [26, 27, 22]. Broadly speaking, communications in MARL can be divided into two main focuses: parameter sharing, which uses a single neural network whose parameters are shared among all agents [27], and memory driven learning, in which the agents use shared memory as a communication channel as a way to first read the memory and then write the responses [28].

3.3 Cooperation and Competitions

Although explicit communication can be used in cooperation, it is not a necessity for cooperative problems. In this category, the analyzed works are evaluated in either fully cooperative or fully competitive or mixed settings.

3.3.1 Fairness

When we extend to multiple agents, we may not always want to learn the "best arm" since this notion may not be accurate or fair in this setting. This is because

each agent may perceive a different arm to be the best for itself. Therefore, the goal becomes to learn a fair distribution over the arms.

[29] uses Nash social welfare, a fairness objective borrowed from computational social choice, to make a fair collective decision. A distribution p that places probability p_j on each arm j gives expected utility $p_j \cdot \mu_{i,j}$ to agent i. The objective for Nash social welfare is:

$$NSW(p, \mu^\star) = \prod_{i=1}^{N} \left(\sum_{j=1}^{K} p_j \cdot \mu_{i,j}^\star \right)$$

Hence, according to this criterion, the fairest distribution maximizes the product of the expected utilities to the agents.

3.3.1.1 Explore-First The analysis here is similar to that of the single agent setting beginning in the exploration stage (single-agent explore-first), where each agent pulls each arm L times. However, at the end of this stage, we compute a policy $\hat{p}$ with the best estimated Nash social welfare. During exploitation, it uses policy $\hat{p}$ in every round.

The regret bound is also similar to single-agent explore-first ($K^{\frac{1}{3}} T^{\frac{2}{3}} \log^{\frac{1}{3}}(T)$), but with an extra $N^{\frac{2}{3}}$ term.

3.3.1.2 Epsilon-Greedy This is also similar to the single-agent variant where at each round t, exploration is performed with probability ϵ^t in which arms are cycled through in round-robin fashion. Otherwise, the algorithm exploits by using the policy $\hat{p}_t$ with the highest Nash social welfare under the current estimated reward matrix.

Epsilon-Greedy is horizon-independent, and has an expected regret of :
$N^{\frac{2}{3}} K^{\frac{1}{3}} T^{\frac{2}{3}} \log^{\frac{1}{3}}(NKt)$ for $\epsilon^t = N^{\frac{2}{3}} K^{\frac{1}{3}} t^{-\frac{1}{3}} \log^{\frac{1}{3}}(NKt)$

3.3.1.3 UCB This multi-agent version of UCB similarly selects a policy that maximizes the estimated Nash social welfare plus a confidence term. This confidence term involves taking a linear combination of each agent's confidence interval.

If $\alpha^t = N \ \forall t$, the expected regret is $NKT^{\frac{1}{2}} \log(NKT)$

3.3.1.4 Distribution Generalization Reward distributions are generally always assumed to be sub-Gaussian, which is a probability distribution with a strong tail decay. However, increasing evidence shows that this may not hold in many applications, some of which include distributed load estimation of internet traffic and multi-agent modeling of supply chain networks. In coordinated tasks, this also applies to situations with a non-zero bias in the communication channel.

[30] introduces robust cooperative algorithms that are studied on heavy-tailed random variables, which are variables that do not admit a finite moment-generating function.They derive an algorithm MP-UCB for the cooperative

multi-agent stochastic bandit under heavy-tailed densities. The main technique is to control the variance of the arm estimators across the communication graph G when the consensus protocol provides suboptimal guarantees. This is done by incorporating robust mean estimators to achieve optimal rates.

They derive a lower bound of $K\Delta^{-\frac{1}{\epsilon}\ln(T)}$ on the group regret.

3.3.2 MARL with Cooperations and Competitions

Many recent MARL works build upon the game theoretical environment called *Sequential Social Dilemma (SSD)*, a framework described in *section 2.1.2.2* [5]. MARL algorithms with emergent behaviors tend to have cooperation-like and competition-like behaviors [5, 8, 7, 21, 22]. Notice that the SSD framework defines cooperations and defections not on the basis of individual actions, but rather on overall policy trajectories. Some of the works either focus entirely on cooperation [8, 22], while some other focused on mixed strategies [5, 31, 7].

3.4 Inference of Other Agents

These works explicitly model other agents, thereby helping with goal inference and accounting for other agents' learning behavior.

3.4.1 Robustness

By inferring the behavior of specialized agents, multi-agent algorithms can become more robust to various conditions.

3.4.1.1 Malicious Agent [32] generalizes the original multi-agent bandit setting to include n honest and m malicious agents. This is particularly applicable to many real-life scenarios, such as identifying machine faults in a distributed system or spam in a social recommendation system.

They present a method in which honest agents learn which agents are malicious, and then dynamically reduce communication with them. This is done through "blocking", meaning that if an arm recommended by a particular other agent performs poorly at timestep t, the current agent will ignore that agent's recommendations until step t^2. By blocking in this interval, we prevent over-penalization of honest agents who mistakenly recommend bad arms at small t due to noise in the environment. Simultaneously, malicious agents who repeatedly recommend bad arms are punished with increasing severity.

The upper bound for the regret is upper bounded by $(m + k/n)log(T/\Delta)$, where Δ is the arm gap.

3.4.2 Inference of Other Agents in MARL

Several works in MARL focus on using causality to infer other agents' intentions [8, 33, 31, 7, 21]. The works in this category can be divided into those using behavioral metrics such as social influence [33] and inequity aversion [8], and

those that focus on joint policy optimizations [31, 7, 22], which overlap with cooperation-focused methods mentioned in *section 3.3.2*.

4 Conclusion and Discussion

We present a literature review for both the multi-armed bandit and reinforcement learning problem and grouping their respective multi-agent learning algorithms under common themes of:

- Emergent behaviors
- Communication
- Cooperation and Competition
- Inference of Other Agents

In general, these categorizations had more of an overlap in the bandit literature, on top of which there weren't works relating to emergent behaviors mainly due to the nature of the bandit problem being more constrained and applicable to a smaller subset of tasks. Reinforcement learning, although more general and more readily used for a wide variety of problems, these papers usually don't provide as much of a theoretical justification. We hope that this report work will serve as a reference to mutually gain insights on both the reinforcement learning and bandit problems and provide inspiration when thinking of new methods to approach either one.

References

[1] W. R. Thompson, "On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples," *Biometrika*, vol. 25, no. 3-4, pp. 285–294, 12 1933.

[2] S. Agrawal and N. Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," *CoRR*, vol. abs/1111.1797, 2011.

[3] E. Bargiacchi, T. Verstraeten, D. Roijers, A. Nowe, and H. Van Hasselt, "Learning to coordinate with coordination graphs in repeated single-stage multi-agent decision problems," 07 2018.

[4] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 157–163.

[5] J. Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel, "Multi-agent Reinforcement Learning in Sequential Social Dilemmas," *arXiv:1702.03037 [cs]*, Feb. 2017, arXiv: 1702.03037.

[6] P. J. Gmytrasiewicz and P. Doshi, "A Framework for Sequential Planning in Multi-Agent Settings," *Journal of Artificial Intelligence Research*, vol. 24, pp. 49–79, Jul. 2005.

[7] R. Köster, K. R. McKee, R. Everett, L. Weidinger, W. S. Isaac, E. Hughes, E. A. Duéñez-Guzmán, T. Graepel, M. Botvinick, and J. Z. Leibo, "Model-free conventions in multi-agent reinforcement learning with heterogeneous preferences," *arXiv:2010.09054 [cs]*, Oct. 2020, arXiv: 2010.09054 version: 1.

[8] E. Hughes, J. Z. Leibo, M. G. Philips, K. Tuyls, E. A. Duéñez-Guzmán, A. G. Castañeda, I. Dunning, T. Zhu, K. R. McKee, R. Koster, H. Roff, and T. Graepel, "Inequity aversion resolves intertemporal social dilemmas," *CoRR*, vol. abs/1803.08884, 2018.

[9] L. Panait and S. Luke, "Cooperative Multi-Agent Learning: The State of the Art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, Nov. 2005.

[10] Y. Shoham, R. Powers, and T. Grenager, "If multi-agent learning is the answer, what is the question?" *Artificial Intelligence*, vol. 171, no. 7, pp. 365–377, May 2007.

[11] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[12] P. Stone, "Multiagent learning is not the answer. it is the question," *Artificial Intelligence*, vol. 171, pp. 402–05, May 2007.

[13] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. de Cote, "A Survey of Learning in Multiagent Environments: Dealing with Non-Stationarity," p. 64.

[14] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, Nov. 2019.

[15] S. V. Albrecht and P. Stone, "Autonomous agents modelling other agents: A comprehensive survey and open problems," *CoRR*, vol. abs/1709.08071, 2017. [Online]. Available: http://arxiv.org/abs/1709.08071

[16] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017.

[18] R. S. Sutton and A. G. Barto, *Reinforcement Learning, Second Edition*. The MIT Press, 2018.

[19] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[20] J. Z. Leibo, E. Hughes, M. Lanctot, and T. Graepel, "Autocurricula and the Emergence of Innovation from Social Interaction: A Manifesto for Multi-Agent Intelligence Research," *arXiv:1903.00742 [cs, q-bio]*, Mar. 2019, arXiv: 1903.00742.

[21] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch, "Emergent Tool Use From Multi-Agent Autocurricula," *arXiv:1909.07528 [cs, stat]*, Feb. 2020, arXiv: 1909.07528.

[22] B. Baker, "Emergent Reciprocity and Team Formation from Randomized Uncertain Social Preferences," p. 14.

[23] P. Landgren, V. Srivastava, and N. E. Leonard, "On distributed cooperative decision-making in multiarmed bandits," 2019.

[24] T. Verstraeten, E. Bargiacchi, P. Libin, D. Roijers, J. Helsen, and A. Nowe, "Multi-agent thompson sampling for bandit applications with sparse neighbourhood structures," *Scientific Reports*, vol. 10, p. 6728, 04 2020.

[25] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 151–159.

[26] S. Sukhbaatar, A. Szlam, and R. Fergus, "Learning multiagent communication with backpropagation," *CoRR*, vol. abs/1605.07736, 2016.

[27] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 2145–2153.

[28] E. Pesce and G. Montana, "Improving coordination in multi-agent deep reinforcement learning through memory-driven communication," *CoRR*, vol. abs/1901.03887, 2019. [Online]. Available: http://arxiv.org/abs/1901.03887

[29] S. Hossain, E. Micha, and N. Shah, "Fair algorithms for multi-agent multi-armed bandits," 2021.

[30] A. Dubey and A. Pentland, "Cooperative multi-agent bandits with heavy tails," 2020.

[31] A. S. Vezhnevets, Y. Wu, R. Leblond, and J. Z. Leibo, "Options as responses: Grounding behavioural hierarchies in multi-agent RL," *CoRR*, vol. abs/1906.01470, 2019.

[32] D. Vial, S. Shakkottai, and R. Srikant, "Robust multi-agent multi-armed bandits," 2020.

[33] N. Jaques, A. Lazaridou, E. Hughes, Ç. Gülçehre, P. A. Ortega, D. Strouse, J. Z. Leibo, and N. de Freitas, "Intrinsic social motivation via causal influence in multi-agent RL," *CoRR*, vol. abs/1810.08647, 2018. [Online]. Available: http://arxiv.org/abs/1810.08647