Algorithms in Multi-Agent Systems: A Holistic Perspective from Reinforcement Learning and Game Theory

Lu Yunlong
luyunlong@pku.edu.cn
School of EECS, Peking University

Yan Kai
289371298@pku.edu.cn
School of EECS, Peking University

Abstract

Deep reinforcement learning has achieved state-of-the-art performance in single-agent games, with a drastic development in its methods and applications. Recent works are exploring its potential in multi-agent scenarios. However, they are faced with lots of challenges and are seeking for help from traditional game-theoretic algorithms, which, in turn, show bright application promise combined with modern algorithms and boosting computing power. In this survey, we first introduce basic concepts and algorithms in single-agent RL and multi-agent systems; then, we summarize the related algorithms from three aspects. Solution concepts from game theory give inspiration to algorithms which try to evaluate the agents or find better solutions in multi-agent systems. Fictitious self-play becomes popular and has a great impact on the algorithm of multi-agent reinforcement learning. Counterfactual regret minimization is an important tool to solve games with incomplete information, and has shown great strength when combined with deep learning.

1. Introduction

No man is an island entire of itself. We live in a world where people constantly interact with each other, and most situations we confronted with every day involve cooperation and competition with others. Though recent years have witnessed dramatic progress in the field of AI, its application will still be limited until intellectual agents have learnt how to cope with others in a multi-agent system. The first attempt of defining multi-agent systems (MAS) dates back to 2000, when Stone and Veloso defined the area of MAS and stated its open problems [1], and the number of algorithms and applications in this area has been rising ever since. In the last decade, multi-agent learning has achieved great success in games which were once considered impossible for AI to conquer, including games with tremendous number of states like Go [2], Chess and Shogi [3], games with incomplete information like Texas Hold'em [4] and Avalon [5], and multi-player video games with extremely high complexity like Dota 2 [6] and StarCraft [7].

The main reason for the current success in multi-agent games is the combination of techniques from two main areas: deep reinforcement learning and game theory. The former provides powerful algorithms for training agents with a particular goal in an interactive environment, but it cannot be straightforwardly applied to a multi-agent setting [8]; the latter is born to analyze the behavior of multiple agents, but is developed mainly in theory, with algorithms capable of solving problems in a very small scale.

Deep reinforcement learning (DRL) is the combination of reinforcement learning (RL) and deep learning. RL [9] is an area of machine learning concerned with how agents ought to take actions in an environment in order to maximize some notion of cumulative rewards. Deep learning [10] is a class of machine learning algorithms that uses neural networks (NN) to extract high-level features from the raw input. Before the prevalence of deep learning, RL needs manually-designed features to represent state information when it comes to complex games; neural network can serve as an adaptive function approximator, allowing RL to scale to problems with high-dimensional state space [11] and continuous action space [12] [13] [14].

By adopting such approximator, DRL is proved to be successful in single-player games with high-dimensional input and large state space like Atari [15]. However, straightforward application of DRL to multi-agent scenarios only achieved limited success both empirically [16] and theoretically [17]. Modeling other agents as part of the environment makes the environment adversarial and no longer Markov, violating the key assumption in the original theory of RL [9].

Therefore, a mature approach has to consider carefully the nature of a multi-agent system. Compared with single-agent game where the only agent aims to maximize its own payoff, in a multi-agent system the environment includes other agents, all of whom aim to maximize their payoffs. There are special cases such as potential games or fully co-

operative games where all agents can reach the same global optimal; however, under most circumstances an optimal strategy for a given agent does not make sense any more, since the best strategy depends on the choices of others. In game theory [18], researchers are focused on certain solution concepts, like some kind of equilibrium, rather than optimality [19].

Nash equilibrium [19] is one of the most fundamental solution concepts in game theory, and is widely used to modify single-agent reinforcement learning to tackle multi-agent problems. However, Nash equilibrium is a static solution concept based solely on fixed point, which is limited in describing the dynamic properties of a multi-agent system like recurrent sets, periodic orbits, and limit cycles [20] [21]. Some researchers try to establish new solution concepts more capable of describing such properties, and use them to evaluate or train agents in a multi-agent system. The attempts include bounded rationality [22] and dynamic-system-based concepts such as Markov-Conley Chain (MCC), proposed by α-rank [21] and applied in [23] and [24].

In addition to focusing on solution concepts, some game-theoretic algorithms have sparked a revolution in computer game playing of some of the most difficult games.

One of them is called is called fictitious self-play [25], which is a sample-based variant of fictitious play [26], a popular game-theoretic model of learning in games. In this model, players repeatedly play a game, at each iteration choosing a best response to their opponents' average strategies. Then the average strategy profile of fictitious players converges to a Nash equilibrium in certain classes of games. This algorithm has laid the foundation of learning from self-play experiences and has a great impact on the algorithm of multi-agent reinforcement learning, producing many exciting results including AlphaZero [3] when combined with enough computing power.

Another one of them is called counterfactual regret minimization [27], which is based on the important game-theoretic algorithm of regret matching introduced by Hart and Mas-Colell in 2000 [28]. In this algorithm, players reach equilibrium play by tracking regrets for past plays, making future plays proportional to positive regrets. This simple and intuitive technique has become the most powerful tool to deal with games with incomplete information, and has achieved great success in games like Poker [4] when combined with deep learning. In this survey, we will focus on the above two algorithms and how they are applied in reinforcement learning, especially in competitive multi-agent RL.

This survey consists of four parts: Section 2 includes the basic concepts and algorithms in single-agent reinforcement learning, as well as basic concepts of multi-agent system; Section 3 mainly describes the inspiration that solution concepts give to multi-agent RL; Section 4 states how fictitious self-play grows into an important tool for multi-agent RL; Section 5 gives an introduction for counterfactual regret minimization and its application in multi-agent RL.

2. Background

2.1. Single-Agent Reinforcement Learning

One of the simplest forms of single-agent reinforcement learning problems is the multi-armed bandit problem [29], where players try to earn money (maximizing *reward*) by selecting an arm (an *action*) to play (*interact* with the environment). This is an oversimplified model, where the situation before and after each turn is exactly the same. In most scenarios, you may face different conditions and take multiple actions in a row to complete one turn. This introduce the discrimination between *states* and their *transitions*.

More formally, a single-agent RL problem can be described as a Markov Decision Process (MDP) [9], which is a quintet $< S, A, R, T, \gamma >$ where S represents a set of states and A represents a set of actions. The transition function $T : S \times A \to \Delta(S)$ maps each state-action pair to a probability distribution over states. Thus, for each $s, s' \in S$ and $a \in A$, the function T determines the probability of going from state s to s' after executing action a. The reward function $R : S \times A \times S \to \mathbb{R}$ defines the immediate reward that an agent would receive when transits from state s to s' via action a; sometimes the reward can also be a probability distribution on $\mathbb{R}$. $\gamma \in [0, 1]$ defines the discount factor to balance the trade-off between the reward in the next transition and rewards in further steps, and the reward gained k steps later are discounted by a factor of γ^k in the accumulation.

The solution of MDP is a policy $\pi : S \to \Delta(A)$, which maps states to a probability distribution of actions, indicating how an agent chooses actions when faced with each state. The goal of RL is to find the *optimal policy* π^* to maximize the expected discounted sum of rewards. There are different techniques for solving MDPs assuming all of its components are known. One of the most common techniques is the value iteration algorithm [9] based on the Bellman equation:

$$v_\pi(s) = \mathbb{E}_{a\sim\pi(s)}\mathbb{E}_{s'\sim T(s,a)}[R(s, a, s') + \gamma v_\pi(s')] \quad (1)$$

This equation expresses the value (expected payoffs) of a state under policy π, which can be used to obtain the optimal policy $\pi^* = argmax_\pi v_\pi(s)$ i.e. the one that maximizes the value function.

Value iteration is a model-based RL algorithm since it requires a complete model of the MDP. However, in most cases an environment is either not fully known or too complicated. For this reason, model-free algorithms are more

preferred, which learn from experiences of interacting with the environment in a sample-based fashion.

There are two kinds of model-free RL algorithms, value-based and policy-based. Value-based algorithms optimize the policy by tracking the values of states or state-action pairs, and choosing better actions according to these values. Q-learning is one of the most well-known of them. A Q-learning agent learns a Q-function, which is the estimate of the expected payoff starting in state s and taking action a as $Q(s,a)$. Whenever the agent transits from state s to s' by taking action a with reward r, the Q table is updated as:

$$Q(s,a) = Q(s,a)+\alpha[(r+\gamma max_b Q(s',b))-Q(s,a)] \quad (2)$$

where α is the learning rate. It is proved that Q-learning can converge to the optimal Q^* if each state-action pair is visited infinitely often under specific parameters [30] [31]. The most famous value-based method in DRL is the Deep Q-network (DQN) [11], which introduces deep learning into RL by approximating the Q function with a deep NN, allowing RL to deal with problems with high-dimensional state space like pixel space. Moreover, since the neural networks are capable of extracting features by themselves, manual feature engineering with prior knowledge is no longer necessary.

Policy-based algorithms take another route, which directly learn parameterized policies based on gradients of some performance measures using gradient descent method. One of the earliest work is REINFORCE [32], which samples full episode trajectories with Monte Carlo methods to estimate return. The policy parameters θ, where $\pi(a|s,\theta) \approx \pi(a|s)$, is updated as:

$$\theta_{n+1} = \theta_n + \alpha G_n \frac{\nabla \pi(A_n|S_n,\theta_n)}{\pi(A_n|S_n,\theta_n)} \quad (3)$$

where α is the learning rate and G is the discounted sum of rewards. However, pure policy-based methods can have high variance [9] and actor-critic algorithms [33], a combination of value-based and policy-based methods, have been proposed. They use actors to learn parameterized policies and critics to learn value functions, which allows the policy updates to take into consideration the value estimates to reduce the variance compared to vanilla policy-based methods.

Combining policy-based methods with deep learning is straightforward since the parameterized policy can be directly replaced with a deep NN, and there are many variants of them. Deep deterministic policy gradient (DDPG) [12] addresses the issue of large action space by adding sampled noise, such as noise drawn from Ornstein-Uhlenbeck process [34], to its actor's policy, allowing more exploratory behavior. Asynchronous Advantage Actor-Critic (A3C) [35] is a distributed algorithm where multiple actors running on different threads interact with the environment simultaneously and compute gradients in a local manner. UNREAL framework [36] is based on A3C and proposes unsupervised auxiliary tasks like reward prediction to accelerate the learning process. Importance Weighted Actor-Learner Architecture (IMPALA) [37] is another distributed algorithm that allows trajectories of experience to be communicated between actors and a centralized learner. Trust Region Policy Optimization (TRPO) [13] and Proximal Policy Optimization (PPO) [14] are state-of-the-art policy-based DRL algorithms, where changes in policy are incorporated to the loss function by adding KL-divergence to the loss to prevent abrupt changes in policies during training.

2.2. Markov Game and Nash Equilibrium

While MDP models single-agent game with discrete time, the framework of Markov games, or stochastic games [38] , models multi-agent systems with discrete time and non-cooperative nature. An n-player Markov game is defined [39] by a tuple $< S, A^1, ..., A^n, r^1, ..., r^n, p >$, where s is the state space, A^i is the action space of player i, $r^i : S \times A^1 \times ... \times A^n \rightarrow R$ is the payoff function for player i, $p : S \times A^1 \times ... \times A^n \rightarrow \Delta(S)$ is the transition probability map, where $\Delta(S)$ is the set of probability distributions over state space S. Given state s, agents independently choose actions $a^1, ..., a^n$, and receive rewards $r^i(s, a^1, ..., a^n)$, $i = 1, ..., n$. The state then transits to the next state based on the transition probabilities. Specifically, in a discounted Markov game, the objective of each player is to maximize the discounted sum of rewards, with discount factor $\gamma \in [0, 1)$. Let π^i be the strategy of player i, for a given initial state s, player i tries to maximize

$$v^i(s, \pi^1, ..., \pi^n) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}(r_t^i|\pi^1, ..., \pi^n, s_0 = s) \quad (4)$$

Nash equilibrium is one of the most fundamental solution concepts in game theory. It is a joint strategy profile where each agent's strategy is a best response of the others'. Specifically, for a Markov game, a Nash equilibrium is a tuple of n strategies $(\pi_*^1, ..., \pi_*^n)$ such that for all state $s \in S$ and $i = 1, ..., n$,

$$v^i(s, \pi_*^1, ..., \pi_*^n) \geq v^i(s, \pi_*^1, ..., \pi_*^{i-1}, \pi^i, \pi_*^{i+1}, ..., \pi_*^n) \quad for\ all\ \pi^i \in \Pi^i \quad (5)$$

where Π^i is the set of all strategies available to agent i.

In general, the strategies that constitute a Nash Equilibrium can be either stationary strategies or behavior strategies which allow conditioning of actions on history of play. In 1964 Fink [40] proved that every n-player discounted stochastic game possesses at least one Nash equilibrium point in stationary strategies, as a enhanced conclusion of Nash's theorem in [41].

3. Solution Concepts

3.1. Without Solution Concepts: Independent RL

Although algorithms for single-agent reinforcement learning are not designed for multi-agent systems because the assumptions from which they are derived are invalid, there are a group of works studying and analyzing the behaviors directly using independent DRL agents in multi-agent settings, modeling other agents as part of the environment.

Tampuu *et al*. [42] trained two independent DQN learning agents to play the Pong game. They tried to adapt the reward functions to achieve either cooperative or competitive settings. Later, Leibo *et al*. [43] also tried independent DQNs but in the context of sequential social dilemmas. This work showed that cooperative or competitive settings can not only affect discrete actions, but also change the whole policies of agents.

Recently, Bansal *et al*. [44] trained independent learning agents with PPO using the MuJoCo simulator [45]. They applied two modifications to deal with the multi-agent nature of the problem though. First, they used exploration rewards which are dense rewards to allow agents to learn basic, non-competitive behaviors, and reducing this type of reward through time, giving more weight to the environmental, competitive reward. Second, they maintained a collection of older versions of the opponent to sample from, rather than always using the most recent version, to stabilize the behavior changes over the training process.

Raghu *et al*. [16] investigated how different DRL algorithms, including DQN, A2C and PPO, performed in a family of two-player zero-sum games with tunable complexity, called Erdos-Selfridge-Spencer games, which is a parameterized family of environments and the optimal behavior can be completely characterized. Their work showed that different DRL algorithms can show wide variation in performance as the game's difficulty is tuned.

3.2. RL towards Nash Equilibrium

To tackle multi-agent problems, directly using single-agent reinforcement learning proves invalid, and efforts are made to better understand the nature of multi-agent systems. The first step is taken by Littman in 1994 [46], who pointed out that for many Markov games, there is no policy that is undominated because performance depends critically on the choice of opponent. Instead, each policy should be evaluated with respect to the opponent that makes it look the worst. He proposed Minimax-Q in the setting of two-player zero-sum Markov game, where the opponent is not modeled as part of the environment but one that rationally makes your policy worse. Minimax-Q is a modification of single-agent Q-learning, where the original $Q(s,a)$ is substituted with $Q(s,a,o)$, representing the expected payoff for taking action a when the opponent chooses action o from state s and continuing optimally thereafter. The updating rule becomes:

$$\begin{aligned} Q(s,a,o) &= Q(s,a,o) + \alpha[(r + \gamma V(s'))) - Q(s,a,o)] \\ where\ V(s) &= \max_{\pi \in \Delta(A)} \min_{o \in O} \mathbb{E}_{a \sim \pi} Q(s,a,o) \end{aligned} \tag{6}$$

Then in 2001, he proposed Team-Q [47] to tackle another special case of Markov games called team Markov game, where agents share the same reward function i.e. have the same objective. In this case, $Q_i(s, a^1, ..., a^n)$ is the Q-function of agent i and the updating rule becomes:

$$\begin{aligned} Q_i(s,\vec{a}) &= Q_i(s,\vec{a}) + \alpha[(r + \gamma V_i(s'))) - Q_i(s,\vec{a})] \\ where\ V_i(s) &= \max_{\vec{a} \in A(s)} Q_i(s,\vec{a}) \end{aligned} \tag{7}$$

In this work, he pointed out that Minimax-Q and Team-Q can be considered as two special cases of a more general algorithm called Nash-Q, because both algorithms are updating the Q-function with values from Nash equilibrium, either in cooperative or competitive scenarios. In the same year, he unified them into Friend-and-Foe-Q [48], where cooperation equilibrium and adversarial equilibrium can coexist in a multi-agent system, and the update of Q-function is unified into:

$$\begin{aligned} &Q_i(s,\vec{a}) = Q_i(s,\vec{a}) + \alpha[(r + \gamma V_i(s'))) - Q_i(s,\vec{a})] \\ &\text{where } V_i(s) = \max_{\pi \in \Delta(X_1) \times ... \times \Delta(X_k)} \min_{y_1,...,y_l \in Y_1 \times ... \times Y_l} \\ &\qquad \mathbb{E}_{x_1,...,x_k \sim \pi} Q_i(s, x_1, ..., x_k, y_1, ..., y_l) \end{aligned} \tag{8}$$

where $X_1, ..., X_k$ are the actions available to the k friends of player i and $Y_1, ..., Y_l$ are those of l foes, based on the idea that i's friends are working together to maximize i's value, while i's foes are working together to minimize i's value.

Although Littman had suggested a general Nash-Q algorithm in [47], it was not until Hu *et al*. [49] that a formal formulation of Nash-Q and poof of convergence were proposed. In this work, the Q-function is updated by:

$$Q_i(s,\vec{a}) = Q_i(s,\vec{a}) + \alpha[(r + \gamma Nash_i(s'))) - Q_i(s,\vec{a})] \tag{9}$$

where $Nash_i(s)$ is agent i's payoff in state s in the selected Nash equilibrium. It is proved that Nash-Q converges under a strict condition that, every stage game during learning has either a global optimal or a saddle point, and they are always selected to update the Q-function, where a saddle point means a Nash equilibrium where each agent would receive a higher payoff when at least one of the other agents deviates.

Shortly after, multiple variants of Nash-Q were proposed in a very similar fashion. Greenwald [50] used a general form of Nash equilibrium called correlated equilibrium

[51], which is a probability distribution over the joint space of actions, where all agents optimize with respect to others' probabilities, conditioned on their own. He proposed Correlated-Q and its four variants focused on different correlated equilibrium, as an attempt to unify the previous works:

1. maximize the sum of the players' rewards
2. maximize the minimum of the players' rewards
3. maximize the maximum of the players' rewards
4. maximize the maximum of each individual player i's rewards

In the same year, Könönen [52] introduced Asymmetric-Q to deal with Stackelberg leadership model, a game where the leader moves first and the follower move sequentially. Based on the hierarchical equilibrium solution concept called Stackelberg equilibrium [53], Asymmetric-Q uses a updating rule of Q-function similar to Nash-Q, except in a hierarchical learning order.

3.3. Beyond Nash Equilibrium

Nash Equilibrium proves to be powerful in several specific types of games, but it has never been compatible with general-sum games with more than two players. When Nash equilibrium is applied to reinforcement learning, small changes in the values of joint-actions may cause a large change in the state's Nash equilibria, making the training unstable and unable to converge [54]. There are attempts, both theoretically and experimentally, to bypass the incompatibility by either redefining the equilibrium [55] or straightforwardly applying the algorithm for two-player zero-sum games to general-sum games with more than two players [56], but these attempts only provide limited success in specific scenarios.

Another problem is that Nash equilibrium is notoriously hard to compute. Chen and Deng [57] proved that calculating Nash equilibrium for general-sum games is PPAD-complete for two players, which requires exponential-time algorithm. Even if all the equilibria are found, there is no guarantee which equilibrium an algorithm would fall in; it may probably find an ineffective equilibrium with less payoff for both agents. There are researches about other kinds of equilibrium such as Stackelberg equilibrium [53] [52]. However, these equilibrium can only deal with specific types of problems, far less useful than Nash equilibrium defined for general scenarios. Therefore, current studies about game theory in RL are still limited in the few types of games Nash can solve, including two-player zero-sum games and team Markov games.

Recently, researchers from DeepMind proposed an algorithm called α-rank [21] for ranking the performance of different agents in a multi-agent system. In this work, they introduced a new solution concept called Markov-Conley Chain (MCC), based on strongly sink connected components on transition graphs of Markov chains, which can be seen as a discrete version of dynamic systems in continuous space. According to the Conley's theorem [58], the domain of any dynamic system can be decomposed into its chain components with the remaining points transient, leading to the recurrent part by a complete Lyapunov function. Therefore, under the solution concept based on dynamic systems, there exists a potential function, which is a complete Lyapunov function, as the incentive of optimization for all agents. Such games, called potential games, have Nash equilibrium at local optima, thus greatly simplifying the process of finding equilibrium.

α-rank is inspired by evolutionary algorithms. Consider a monomorphic population wherein all individuals play identical strategies, and a monomorphic population profile consisting multiple populations where each population may be playing a different strategy. Mutation rate is fixed to a very small number so that each time there is at most one individual in one population that is mutated. Individuals are constantly chosen from each population to involve in the game and be evaluated and those with better performance have better chance to reproduce. So the only mutation will either spread until taking over the whole population, or be wiped out.

Since the result of mutation only ends up in changing behavior of the whole population or not, this process can be viewed as a random walk of the whole monomorphic population profile on states of different strategy profiles, where adjacent states differ by exactly one agent's strategy. This graph is called "response graph". More specifically, the transitional matrix of this Markov process can be written as:

$$\rho^k_{s_i^k, s_j^k}(s^{-k}) = \begin{cases} \dfrac{1 - e^{-\alpha(r_i^k - r_j^k)}}{1 - e^{-m\alpha(r_i^k - r_j^k)}} & \text{if } r_i^k \neq r_j^k \\ \dfrac{1}{m} & \text{if } r_i^k = r_j^k \end{cases} \tag{10}$$

$$C_{i,j} = \begin{cases} \eta \rho^k_{s_i^k, s_j^k}(s^{-k}) \text{ if } \exists k \text{ such that } s_i^k \neq s_j^k \text{ and } s_i^{-k} = s_j^{-k} \\ 1 - \sum_{j \neq i} C_{i,j} \text{ if } s_i = s_j \\ 0 \qquad\qquad \text{otherwise} \end{cases} \tag{11}$$

where $C_{i,j}$ is the transition probability from strategy profile i to profile j, r_i^k and r_j^k are the rewards of agent k in profile i and j, s_i^k is k-th player's strategy in profile i and s_i^{-k} is others' strategies in profile i, η is the reciprocal of adjacent strategy profile, α is a hyper-parameter standing for the harshness of natural selection and m is the number of individuals in each population.

After constructing such Markov chain, its stationary distribution can be calculated and the larger probability a profile has, the stronger and more robust it is, which can be

used to rank different strategy profiles. When α goes to infinity, all probability on the graph will eventually converge to the sink strong connected component of the directed response graph, where one profile is connected to another if and only if there exists a player with utility improvement. Therefore, the complete Lyapunov function guaranteed by Conley's theorem serves as an approximate potential function on the response graph, which is the theoretical basis of the algorithm.

The invention of a new solution concept and its corresponding ranking algorithm provides a powerful polynomial-time tool for strategy analysis of multi-agent general-sum games, bypassing the limitations of Nash equilibrium and surpassing Elo, which can perform badly when there is no total-order relations or there is a large population of low-level agents. Later, Torreno *et al*. [59] generalizes this algorithm to partially observable environments. However, researchers from Huawei [24] claim that α-rank is impractical, for they use strategy profiles to conceal the exponential essence of their algorithms with respect to the number of strategies and agents. Instead, they propose a scalable alternative called α^{α}-rank based on MCC, which is capable of evaluating tens of millions of strategy profiles. Their key idea is to use sample-based stochastic optimization instead of a linear-algebra-based solution, and solving the stationary distribution is rewritten as an optimization problem to minimize the distance of a vector and the vector multiplied by the distribution matrix in this case.

4. Fictitious Self-Play

4.1. Classical Fictitious Play

Fictitious play is first proposed by Brown in 1951 [26]. In fictitious play, each agent models other agents' average strategies, and play the best response against them. The process is repeated until convergence if possible. It is proved that the original fictitious play can converge to Nash equilibrium for two-player zero-sum games [60], 2*2 games [61], potential games [62], games with an interior ESS [63] and certain classes of super-modular games [64].

However, since best response is generically pure, a player's choices under fictitious play are sensitive to the values of the opponents' average strategy, and experiments can never converge to a mixed Nash equilibrium (even if players have a mixed best response in mind). Worse still, fictitious play does not provide convergence guarantee for general sum games. Therefore, Fudenberg and Kreps introduced stochastic fictitious play [65]. Their setting assumes that in a standard game where players move simultaneously, each player privately observes a small noise on their payoffs. This noise turns the original game into a stochastic game and smooths the best response, turning it into a continuous function, as is shown in Fig. 1.

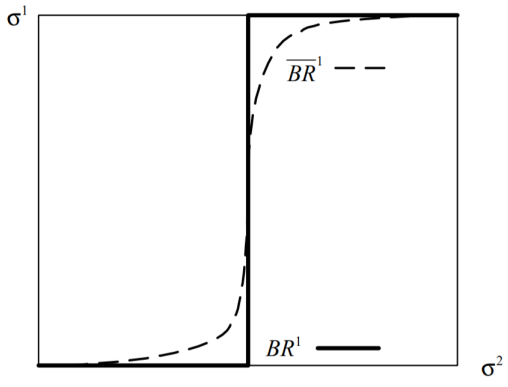

Figure 1. The best response has been "smoothed" into a continuous function by introducing noise. BR stands for best response and δ_1, δ_2 are for two strategies. The picture is from [66].

The benefits are threefold. First, by creating a continuous function, we can correspond the (possibly mixed) equilibrium of the original game with each Nash equilibrium of the perturbed game. Second, such uncertainty prevents the player's strategy from being deduced and exploited by others. Third, this is a model more consistent with actual human psychology [66].

In their work, it is proved that under some assumptions any mixed-strategy equilibrium of the original game is a limit of pure-strategy Bayesian equilibrium of the perturbed game, when the noise approaches zero [66]. Later, Perkins *et al*. [67] further proved that stochastic fictitious play converges to an equilibrium in two-player zero-sum games and its variant converges to an equilibrium in negative-definite single-population games with continuous action sets.

For games that admit a unique Nash equilibrium, the empirical distribution of stochastic fictitious play almost surely converge to the Nash equilibrium. It is later proved by [68] that four special types of games can converge globally: games with an interior ESS (Evolutionary Stable Strategy), zero-sum games, potential games and all super-modular games. Super-modular games are a special type of games where the marginal value of one player's action will increase due to other players' actions.

4.2. RL with Fictitious Self-Play

There are two main obstacles preventing fictitious play from incorporating reinforcement learning. For one thing, fictitious play is an algorithm that requires playing best response to others' average strategies in each stage, while reinforcement learning is a learning algorithm that optimizes the policy, slowly moving to the best response. For another, fictitious play is designed on normal-form games, while reinforcement learning is modeled on MDPs, which can be seen as extensive games with multiple steps. Rewriting an extensive game in matrix form is extremely inefficient

because it can yield exponentially large state and action spaces.

Genugten [69] unintentionally took the first step of combining fictitious play and reinforcement learning, when he proposed weakened fictitious play in two-player zero-sum games, where agents play a better response to others' average policy instead of best response. In this work, he introduced ϵ-best response, the set of response with reward r so that $r \leq R - \epsilon$ when R is the best response, where ϵ is initially large but asymptotically vanishing to 0. Weakened fictitious play is originally invented for speeding up the process of convergence, but it unintentionally opens the door for machine learning algorithms like reinforcement learning, which slowly optimizes the policy towards best response. Later, Leslie *et al*. [70] generalized the setting from two-player zero-sum games to general-sum games with more than two players.

Then in 2015, Heinrich *et al*. [25] proved a critical lemma based on Kuhn's theorem [71] that for a player with perfect recall, any mixed strategy is realization-equivalent to a behavior strategy, and vice versa. Heinrich's lemma gives the method of mixing normal-form strategies by a weighted combination of realization-equivalent behavior strategies. The formal statement is as follows: Let π and β be two behavior strategies, Π and B be two mixed strategies that are realization-equivalent to π and β, and $\lambda_1, \lambda_2 \in R_{\geq 0}$ with $\lambda_1 + \lambda_2 = 1$. Then for each information state $u \in U$,

$$\mu(u) = \pi(u) + \frac{\lambda_2 x_\beta(\delta_u)}{\lambda_1 x_\pi(\delta_u) + \lambda_2 x_\beta(\delta_u)}(\beta(u) - \pi(u)) \quad (12)$$

defines a behavioural strategy μ and μ is realization equivalent to the mixed strategy $M = \lambda_1\Pi + \lambda_2 B$.

This formula expresses mixed strategies in extensive-form games without rewriting them into normal-form games. By using deep reinforcement learning (DQN in his work) and imitation learning for the calculation of best response and average policies, Heinrich proposed Neural Fictitious Self Play (NFSP) [25] in the same paper, marking the beginning of fictitious self-play entering multi-agent reinforcement learning. In 2016, Heinrich generalized NFSP to partially observable environments, achieving superhuman performance on limited Texas Poker [72]. It is worth noting that deep reinforcement learning serves not only as a powerful calculation method, but also as an end-to-end solver eliminating the need to manually design features with prior knowledge in specific fields.

More attempts are made to combine RL algorithms with fictitious play. Zhang *et al*. [73] proposes MC-NFSP, which uses Monte Carlo tree search to further enhance the best strategy. In the tree search, the agents choose action to maximize a weighted sum of Q-function and an exploration reward, thus enhancing the exploration ability of the self-play. The paper also proposes Asynchronous NFSP (ANFSP), which is an extension of original NFSP to allow distributed training. Perolat *et al*. [74] builds a stochastic approximation of the fictitious play process with an architecture inspired by actor-critic algorithms. In this algorithm, the actor and critic updates their policy together towards best response, and it is proved that in a zero-sum two-player multi-state game, this algorithm converges to a Nash equilibrium.

4.3. Applications of Fictitious Self-Play

There are many applications of fictitious self-play in reinforcement learning, mostly related to adversarial training and evolutionary algorithms. Gupta *et al*. [75] use self-play to simultaneously train a cyber attacker and defender. The attacker attempts to deform the input from ground truth to reduce the performance of an RL agent, while the defender functions as a preprocessor for that agent, correcting the deformed input. LOLA algorithm [76], proposed by OpenAI, takes the opponents' gradient into consideration in adversarial training, and achieves fancy results using self-play on complicated environments such as MuJoCo [45]. Kawamura *et al*. [77] empirically combine policy gradient algorithms with NFSP and successfully apply it on a non-trivial RTS game. AlphaGo and its successors [3] combine self-play with deep reinforcement learning and Monte Carlo tree search, achieving state-of-the-art performance on two-player zero-sum games like Go, Chess and Shogi. Sukhbaatar *et al*. [78] use self-play to conduct an unsupervised curriculum learning, letting the agents learn about its environment without giving a reward function.

Double oracle [79] is a well-known variant of fictitious self-play on two-player zero-sum games, which is designed for adapting environment with a reward function correlated with the opponent's behavior. In the algorithm, the "oracle" stands for the mechanism to calculate the best pure strategic response of any mixed strategy of both sides. The algorithm starts from a small set of actions, and both sides calculate the optimal mixed strategy against the previous set. Then, both sides assume that the opponent will take the optimal mixed strategy, and calculate the best pure strategic response against them, which is added to the set. The process is repeated until the set no longer grows. Bošanský [80] extends double oracle to extensive-form games, where the agents choose a sequence of actions from a set of action sequences, and the best response sequence is added for next iteration. Double oracle is proved to converge to a minimax equilibrium for two-player zero-sum games, and has applications in security scenarios [81].

Policy Space Response Oracle (PSRO) [82] is a grand unified model of fictitious play, independent reinforcement learning and double oracle, for all of them are specific instances of PSRO. PSRO is built on meta-game analysis, inherited from empirical game-theoretic analysis (EGTA)

[83]. For every specific game to solve, there exists a meta-game whose action is to choose a policy. A finite set of policies is maintained as a population, which grows as more policies are explored later in training. For each agent in each episode, a set of opponents' policies is sampled from the population and a best response (the oracle) is calculated, which will be included in the population in the next iteration. Final output of the algorithm is some kind of combination of the population. As for fictitious play, the final output is the average policy of the entire population; as for independent reinforcement learning, the output is the last policy of the population. One of the most successful application of PSRO is AlphaStar by DeepMind [7], which achieves professional level in the multi-player video game StarCraft.

As a meta-game solver, PSRO is compatible with high-level solution concept. Recently, there are attempts [23] to combine PSRO with α-rank [21] to get theoretical proof of performance for general-sum games with multiple agents. In a high-level perspective, PSRO iteratively calculates the best response and puts it in the population, which is exactly the process where probabilities "flow" on the response graph of strategy profiles until converging to the stationary distribution, as is described in α-rank. It is proved that PSRO will converge with respect to MCC, i.e. to the unique sink strongly connected components in response graph, on two-player symmetric games and, if the oracle only find strategies that are not in the population, converge on any game. Unfortunately, there is no efficient way to implement such oracle to calculate best response, so MCC is almost no better than Nash equilibrium when it comes to scalability in practical use.

5. Counterfactual Regret Minimization

5.1. Regret Matching

Counterfactual Regret Minimization (CFR) is by far the most successful family of algorithms to solve complicated extensive-form games, and also a strong weapon in the arsenal of multi-agent RL. CFR is based on regret matching algorithm, which is brought forth by Hart and Mas-Colell [28] to adaptively learn a correlated equilibrium by self-play. Intuitively, regret matching is an algorithm where agents analyze their history and update their policy in a "what-if" manner. For example, in a repeated rock-paper-scissor game, when an agent chooses rock and loses to paper, it will consider what would happen had it played paper or scissor, and cope with the opponent who prefers paper better, as is shown in Table 1. However, simply choosing best response is problematic: not only can this be exploited by a clever adversary, but it is also not practical when the calculation of best response is ineffective e.g. requires exponential time. Therefore, regret matching gives a stochastic response where the probability of actions is proportional to their cumulative positive regrets i.e. the gain of payoff had it played this action before. In the example above, the agent will play scissor twice more likely than paper in the next round, while the actions are arbitrarily chosen if there are no positive regret. It is proved that the distribution of the agents' response will converge to the correlated equilibrium when the game is played enough times.

Round	Action	Opponent	CR	ND
0	N/A	N/A	(0, 0, 0)	(1/3, 1/3, 1/3)
1	rock	paper	(0, 1, 2)	(0, 1/3, 2/3)
2	scissor	rock	(1, 3, 2)	(1/6, 1/2, 1/3)
3	rock	scissor	(1, 1, 1)	(1/3, 1/3, 1/3)

Table 1. Regret matching in the rock-paper-scissor game. The cumulative regrets (CR) and next distributions (ND) are of (rock, paper, scissor).

Regret matching can be viewed as an online learning algorithm where the regrets are learned to model other agents' strategies, and the bound of cumulative regret plays an important role in its performance and convergence. Hart and Mas-Colell proved in the original paper [28] that the cumulative regret grows sublinearly with respect to the number of rounds played. Greenwald *et al*. [84] gives the bound of regret for a generalized form of regret matching, where the probability distribution of actions in the next round can be proportional to the polynomial or exponential form of cumulative regret. More theoretical results are done by Orazio *et al*. [85] [86] to use function approximators in regret matching in the context of deep learning. In the literature of reinforcement learning, however, most work is based on the original form [28] where the probability of actions is proportional to cumulative regret.

5.2. CFR, CFR+ and MCCFR

Counterfactual regret minimization (CFR) [87] uses regret matching algorithm to cope with extensive-form games with an exponential number of strategies with respect to steps of the game, and is born to be compatible with partially observable environment. Formally, let $R_i^T(I,a)$ be the regret for player i in round T in information set I choosing action a, $\pi_{-i}^{\sigma^t}(I)$ be the probability of entering I considering the strategies of every player but i in round t, $\sigma^t|_{I\to a}$ be the strategy profile identical to σ^t except that player i always plays a in I. Then the regret can be calculated as:

$$R_i^T(I,a) = \frac{1}{T}\sum_{t=1}^{T} \pi_{-i}^{\sigma^t}(I)(u_i(\sigma^t|_{I\to a}, I) - u_i(\sigma^t, I)) \tag{13}$$

and the agents update their policies for the next round as:

$$R_i^{T,+}(I,a) = \max(R_i^T(I,a), 0) \tag{14}$$

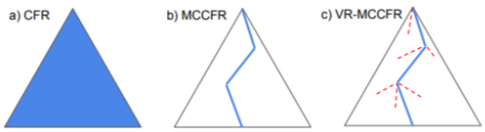

Figure 2. The comparison between CFR, MCCFR and VR-MCCFR. The picture is from [94].

$$\delta_i^{T+1}(I,a) = \begin{cases} \dfrac{R_i^{T,+}(I,a)}{\sum_{a\in A(I)} R_i^{T,+}(I,a)} & \text{if denominator} > 0 \\ \dfrac{1}{|A(I)|} & \text{otherwise.} \end{cases} \tag{15}$$

where $A(I)$ is the action set in information set I. It is proved [87] that the regret bound is $O(1/\sqrt{T})$, and is further improved to $O(1/T^{0.75})$ by Farina *et al*. [88].

In 2014, Tammelin *et al*. [89] [90] brought forth a variant of CFR algorithm called CFR+ by substituting the regret matching algorithm in CFR by regret matching plus. The regret matching plus algorithm calculates the regret as

$$R_i^T(I,a) = R_i^{T-1}(I,a) + v_i(\sigma^T|_{I\to a}, I) - v_i(\sigma^T, I) \tag{16}$$

where the counterfactual value [91] $v_i(\sigma, I)$ is the expectation utility of playing strategy profile σ, which is always 0 unless information set I is reached.

Traditionally, due to the prohibitive time complexity for traversing the whole game tree for extensive-form games, the deployment of CFR in large-scale games such as Limited Texas Hold'em requires prior knowledge and non-trivial work for state abstraction [87] [92]. Monte-Carlo counterfactual regret minimization (MCCFR) [93] is proposed to reduce the time complexity, which is a sample-based variant of CFR. There are two variants of the algorithm: outcome sampling and external sampling, both of which can compute an approximate equilibrium. In outcome sampling, only a single episode of the game is sampled at each iteration, and only the nodes on the path of the game tree in this episode are updated. Since the nodes are sampled according to the opponents' strategy, there is no need to explicitly know the opponents' strategy. External sampling samples only the chance nodes (external to the player) and the opponent's actions, which is proved to make an asymptotic improvement in computational time of equilibrium.

However, Monte Carlo methods are known to be of high variance. To reduce the variance of Monte Carlo method, Schmid *et al*. proposed a variant of MCCFR called VR-MCCFR [94] to speed up the sampling process and reduce the empirical variance. In VR-MCCFR, an action-dependent baseline is defined for each information set in a manner similar to advantage actor-critic (A2C), which allows estimates to bootstrap from other estimates within the same episode while remaining unbiased. The algorithm is also compatible with CFR+ [89], and its comparison with CFR and MCCFR is shown in Figure 2. Later, Johanson *et al*. proposed several sampling methods for MCCFR and MCCFR+ [91]. The core contribution of their work is the proposal of public-chance sampling (PCS) where only the public-chance events (the uncertainty from nature) are sampled. It is proved both theoretically and empirically that PCS can significantly accelerate the training process. There are also attempts to reduce the variance by replacing Monte-Carlo by other sampling methods [95]. However, MCCFR is the basis of most recent works on CFR, and it is the notion of "counterfactual value" that makes Deep CFR possible.

5.3. Deep CFR and its Applications

As the calculation of regret becomes infeasible in large-scale extensive-form games, researchers turn to function approximators for help. The first CFR variant that uses a function approximator is the regression CFR proposed by Waugh *et al*. [96], which uses regression tree as the function approximator. However, this algorithm has two drawbacks. First, handcrafted features about information sets have to be manually designed for the input of the approximator, thus prior knowledge is required. Second, just as vanilla CFR, regression CFR traverses the whole game tree, which makes it unbearably costly with respect to time complexity in non-trivial games.

The milestone that marks CFR eventually joins the family of multi-agent DRL algorithms is the invention of deep counterfactual regret minimization [97]. Deep CFR inherits both the notion of counterfactual value and the sampling method (external sampling in their method) of MCCFR. In this algorithm, a value network is trained to estimate the counterfactual value, whose goal is to approximate the regret value that tabular CFR would have produced. Meanwhile, a policy network is trained to approximate the average strategy played over all iterations, which is optional when there is enough memory to store the policy of every iteration. Later, they propose several weight functions [98] such as linear CFR with weight t, and discounted CFR with weight $\frac{t^\alpha}{t^\alpha+1}$ for iteration t, to put more weight on the recent regret and strategy when calculating cumulative regret and the average strategy. Steinberger [99] simplified deep CFR and reduced the approximation error by abandoning the policy network used to approximate the average strategy.

Many variants of CFR are proposed to better integrate CFR with DRL. Peter *et al*. [100] proposed advantage-based regret minimization (ARM), which maps "information set" I in game theory to "observation" o in reinforcement learning, rewriting the counterfactual action-value pair as $Q_{\pi|o\to a}(o,a)$, which means the agent follows policy π until o is observed and switches to action a. By the

approximation $v_i(\sigma|_{I\to a}, I) = Q_{\pi|o\to a}(o,a) \approx Q_\pi(o,a)$ (notation see section 5.2), the clipped advantage function can be rewritten to express the clipped cumulative regret. So a Q-learning can be conducted to learn the regret value, with actions sampled in the same manner as in original CFR. Li *et al*. [101] proposed double neural CFR, which is another NN-based implementation of CFR. Two LSTM networks are trained in this algorithm, one for estimating the cumulative regret (used to derive the immediate policy) and the other for estimating an average policy.

Besides applications on traditional card games such as Limited Texas Hold'em [102] [4], there are attempts to use deep CFR in games with deception and concealment. Serrino *et al*. [5] trained deep CFR agents to play Avalon, a game where players need to deduct others' identities and hide their own to achieve their goal, and achieved super-human performance with human-explainable agents, though combined with much prior knowledge.

However, as a multi-agent RL algorithm, deep CFR has two limitations. First, despite that CFR is born for partially observable environment, it requires the agent to have perfect recall i.e. recognize everything that happened in the past. Most DRL agents without recurrent neural networks (RNN) like LSTM [103], however, do not have such ability. Second, there are games that never reaches terminal state, and agents are trained to maximize average reward instead. Such environment setting is incompatible with CFR. To solve these problems, Kash *et al*. [104] proposed a Q-learning-style updating rule for agents, where agents uses the CFR algorithm locally in the transition of the Q-learning operator. The algorithm achieves iterative convergence in several games such as NoSDE, a class of Markov games proposed by Littman *et al*. [20] and specifically designed to be challenging to learn, where no prior algorithm converges to an average stationary equilibrium.

6. Conclusion

Deep reinforcement learning has achieved outstanding results in single-agent scenarios [15] in recent years. However, learning in multi-agent systems is fundamentally more difficult and independently using DRL only achieves limited success in specific problems [42] [43] [44] [16]. Game theory helps better understand the nature of multi-agent systems and gives much inspiration to DRL, either in theoretical concepts or algorithms, producing new techniques capable of solving multi-agent games which were once considered impossible [2] [3] [4] [5] [6] [7].

We note that none of these recent exciting results can be achieved by deep reinforcement learning or game-theoretical algorithms alone. However, previous surveys have always tried to contextualize most of multi-agent learning literature into either game theory [105] or deep reinforcement learning [106], which makes it difficult to understand current multi-agent learning algorithms from a holistic perspective.

This survey focuses on algorithms in multi-agent systems derived from both domains, especially their evolutionary history and how they have developed from both domains. We analyze the inspiration that solution concepts give to multi-agent reinforcement learning, how fictitious self-play ends up into the toolbox of RL, and how CFR is combined with deep learning and later unified with DRL. We hope that this survey will incentivize the community of multi-agent learning to see the close connection between the domains of DRL and game theory, and motivate future research in this joint field, taking advantage of the ample and existing literature.

References

[1] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Auton. Robots*, 8(3):345–383, June 2000.

[2] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, October 2017.

[3] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *Science*, 362:1140–1144, 2017.

[4] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, Mar 2017.

[5] Jack Serrino, Max Kleiman-Weiner, David C. Parkes, and Joshua B. Tenenbaum. Finding friend and foe in multi-agent games, 2019.

[6] OpenAI. Openai five. `https://blog.openai.com/openai-five/`, 2018.

[7] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojtek Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis, and David Silver. Alphastar: Mastering the real-time strategy game starcraft ii. `https:`

//deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019.

[8] Tim Baarslag Enrique Munoz de Cote Pablo Hernandez-Leal, Michael Kaisers. A survey of learning in multiagent environments: Dealing with non-stationarity, 2017.

[9] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[11] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[12] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In Yoshua Bengio and Yann LeCun, editors, *ICLR*, 2016.

[13] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR.

[14] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[16] Maithra Raghu, Alex Irpan, Jacob Andreas, Robert Kleinberg, Quoc V. Le, and Jon Kleinberg. Can deep reinforcement learning solve erdos-selfridge-spencer games?, 2017.

[17] Guillaume J. Laurent, Laetitia Matignon, and Nadine Le-Fort Piat. The world of independent learners is not markovian, 2011.

[18] Joel Watson. *Strategy: An Introduction to Game Theory*. W. W. Norton & Company, 2013.

[19] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, Cambridge, UK, 2009.

[20] Martin Zinkevich, Amy R. Greenwald, and Michael L. Littman. Cyclic equilibria in markov games. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, page 1641–1648, 2005.

[21] Shayegan Omidshafiei, Christos Papadimitriou, Georgios Piliouras, Karl Tuyls, Mark Rowland, Jean-Baptiste Lespiau, Wojciech M. Czarnecki, Marc Lanctot, Julien Perolat, and Remi Munos. α-rank: Multi-agent evaluation by evolution, 2019.

[22] Lantao Yu, Jiaming Song, and Stefano Ermon. Multi-agent adversarial inverse reinforcement learning. In *36th International Conference on Machine Learning*, 2019.

[23] Paul Muller, Shayegan Omidshafiei, Mark Rowland, Karl Tuyls, Julien Perolat, Siqi Liu, Daniel Hennes, Luke Marris, Marc Lanctot, Edward Hughes, Zhe Wang, Guy Lever, Nicolas Heess, Thore Graepel, and Remi Munos. A generalized training approach for multiagent learning. In *The 8th International Conference on Learning Representations*, 2019.

[24] Yaodong Yang, Rasul Tutunov, Phu Sakulwongtana, and Haitham Bou Ammar. α^{α}-rank: Practically scaling α-rank through stochastic optimisation, 2019.

[25] Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 805–813. JMLR.org, 2015.

[26] Ulrich Berger. Brown's original fictitious play. Game theory and information, University Library of Munich, Germany, 2005.

[27] Todd W Neller and Marc Lanctot. An introduction to counterfactual regret minimization. In *Proceedings of Model AI Assignments, The Fourth Symposium on Educational Advances in Artificial Intelligence (EAAI-2013)*, 2013.

[28] Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5):1127–1150, 2000.

[29] Michael N. Katehakis and Arthur F. Veinott. The multi-armed bandit problem: Decomposition and computation. *Math. Oper. Res.*, 12:262–268, 1987.

[30] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Oxford, 1989.

[31] John N. Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Mach. Learn.*, 16(3):185–202, September 1994.

[32] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

[33] Vijay R. Konda and John N. Tsitsiklis. Actor-critic algorithms. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 1008–1014. MIT Press, 2000.

[34] G.E.Uhlenbeck and L.S.Ornstein. On the theory of brownian motion. *Phys. Rev.*, 36:823–841, 1930.

[35] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods

for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR.

[36] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *CoRR*, abs/1611.05397, 2016.

[37] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1407–1416, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[38] L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953.

[39] F Thuijsman. *Optimality and Equilibria in Stochastic Games*. Rijksuniversiteit Limburg, 1989.

[40] Arlington M Fink et al. Equilibrium in a stochastic n-person game. *Journal of science of the hiroshima university, series ai (mathematics)*, 28(1):89–93, 1964.

[41] John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.

[42] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PLOS One*, 12, 2015.

[43] Joel Z. Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '17, page 464–473, Richland, SC, 2017. International Foundation for Autonomous Agents and Multiagent Systems.

[44] Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. Emergent complexity via multi-agent competition. In *The 35th International Conference on Machine Learning*, 2017.

[45] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, pages 5026–5033. IEEE, 2012.

[46] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning*, ICML'94, page 157–163, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[47] Michael L. Littman. Value-function reinforcement learning in markov games. *Cogn. Syst. Res.*, 2(1):55–66, April 2001.

[48] Michael L. Littman. Friend-or-foe q-learning in general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 322–328, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[49] Junling Hu and Michael P. Wellman. Nash q-learning for general-sum stochastic games. *J. Mach. Learn. Res.*, 4(null):1039–1069, December 2003.

[50] Amy Greenwald and Keith Hall. Correlated-q learning. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML'03, page 242–249. AAAI Press, 2003.

[51] Robert Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1(1):67–96, 1974.

[52] Ville Könönen. Asymmetric multiagent reinforcement learning. In *Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology*, IAT '03, page 336, USA, 2003. IEEE Computer Society.

[53] Avrim Blum, Nika Hagtalab, MohammadTaghi Hajiaghayi, and Saeed Seddighin. Computing stackelberg equilibria of large general-sum games, 2019.

[54] Michael Bowling. Convergence problems of general-sum multiagent reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, page 89–94, 2000.

[55] Julien Pérolat, Florian Strub, Bilal Piot, and Olivier Pietquin. Learning nash equilibrium for general-sum markov games from batch data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*. PMLR, 2017.

[56] Keigo Kawamura, Naoki Mizukami, and Yoshimasa Tsuruoka. Neural fictitious self-play in imperfect information games with many players. In *Workshop on Computer Games*, 2017.

[57] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of 2-player nash-equilibrium. *Journal of the ACM*, 2007.

[58] Charles Conley. *Isolated invariant sets and the Morse index*, volume 38. R.I.:American Mathematical Society, 1978.

[59] Mark Rowland, Shayegan Omidshafiei, Karl Tuyls, Julien Perolat, Michal Valko, Georgios Piliouras, and Remi Munos. Multiagent evaluation under incomplete information. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, 2019.

[60] Julia Robinson. An iterative method of solving a game. *Annals of Mathematics*, 54:296–301, 1951.

[61] Koichi Miyasawa. On the convergence of learning processes in a 2 x 2 non-zero sum game, 1961.

[62] Dov Monderer and Lloyd S. Shapley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.

[63] J. Hofbauer. From nash and brown to maynard smith: Equilibria, dynamics and ess. *Selection*, 1, 2000.

[64] Sunku Hahn. The convergence of fictitious play in 3 x 3 games with strategic complementarities. *Economics Letters*, 64:57–60, 1999.

[65] Drew Fudenberg and David M. Kreps. Learning mixed equilibria. *Games and Economic Behavior*, 5:320–367, 1993.

[66] Drew Fudenberg and David K. Levine. *The Theory of Learning in Games*. MIT Press, Cambridge, UK, 1998.

[67] S.Perkins and D.S.Leslie. Stochastic fictitious play with continuous action sets. *Journal of Economic Theory*, 152:179–213, 2012.

[68] Josef Hofbauer and William H. Sandholm. On the global convergence of stochastic fictitious play. *Econometrica*, 70:2265–2294, 2002.

[69] Ben Van Der Genugten. A weakened form of fictitious play in two-person zero-sum games. *International Game Theory Review*, 2, 2000.

[70] David S. Leslie and E.J. Collins. Generalised weakened fictitious play. *Games and Economic Behavior*, 56, 2006.

[71] H.W. Kuhn. Extensive games and the problem of information. *Annals of Mathematical Studies*, 28, 1953.

[72] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games, 2016.

[73] Li Zhang, Wei Wang, Shijian Li, and Gang Pan. Monte carlo neural fictitious self-play: Approach to approximate nash equilibrium of imperfect-information games, 2019.

[74] Julien Perolat, Bilal Piot, and Olivier Pietquin. Actor-critic fictitious play in simultaneous move multistage games. volume 84 of *Proceedings of Machine Learning Research*. PMLR, 2018.

[75] Abhishek Gupta and Zhaoyuan Yang. Adversarial reinforcement learning for observer design in autonomous systems under cyber attacks, 2018.

[76] Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *AAMAS '18: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 122–130, 2017.

[77] Keigo Kawamura and Yoshimasa Tsuruoka. Neural fictitious self-play on elf mini-rts, 2019.

[78] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. In *6th International Conference on Learning Representations*, 2018.

[79] H. Brendan McMahan, Geoffery J Gordon, and Avrim Blum. Planning in the presence of cost functions controlled by an adversary. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.

[80] Branislav Bošanský, Christopher Kiekintveld, Viliam Lisý, and Michal Pěchouček. An exact double-oracle algorithm for zero-sum extensive-form games with imperfect information. *J. Artif. Int. Res.*, 51(1):829–866, September 2014.

[81] Manish Jain, Dmytro Korzhyk, Ondrej Vanek, Vincent Conitzer, Michal Pechoucek, and Milind Tambe. A double oracle algorithm for zero-sum security games on graphs. In *In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2011.

[82] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Perolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems 30*, 2017.

[83] William E. Walsh, Rajarshi Das, Gerald Tesauro, and Jeffrey O. Kephart. Analyzing complex strategic interactions in multi-agent systems. In *the 18th National Conference on Artificial Intelligence*, 2002.

[84] Amy Greenwald, Zheng Li, and Casey Marks. Bounds for regret-matching algorithms, 2006.

[85] Ryan D'Orazio, Dustin Morrill, and James R. Wright. Bounds for approximate regret-matching algorithms, 2019.

[86] Ryan D'Orazio, Dustin Morrill, James R. Wright, and Michael Bowling. Alternative function approximation parameterizations for solving games: An analysis of f-regression counterfactual regret minimization, 2019.

[87] Martin Zinkevich, Michael Johanson, and Michael Bowlingand Carmelo Piccione. Regret minimization in games with incomplete information. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, 2007.

[88] Gabriele Farina, Christian Kroer, Noam Brown, and Tuomas Sandholm. Stable-predictive optimistic counterfactual regret minimization. In *Proceedings of the 36th International Conference on International Conference on Machine Learning*, 2019.

[89] Oskari Tammelin. Solving large imperfect information games using cfr+, 2014.

[90] Oskari Tammelin, Neil Burch, Michael Johanson, and Michael Bowling. Solving heads-up limit texas hold'em. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[91] Michael Johanson, Nolan Bard, Marc Lanctot, Richard Gibson, and Michael Bowling. Efficient nash equilibrium approximation through monte carlo counterfactual regret minimization. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, volume 2, pages 837–846, 2012.

[92] Nick Abou Risk and Duane Szafron. Using counterfactual regret minimization to create competitive multiplayer poker

agents. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 159–166, 2010.

[93] Marc Lanctot, Kevin Waugh, Martin Zinkevich, and Michael Bowling. Monte carlo sampling for regret minimization in extensive games. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, pages 1078–1086, 2009.

[94] Martin Schmid, Neil Burch, Marc Lanctot, Matej Moravcik, Rudolf Kadlec, and Michael Bowling. Variance reduction in monte carlo counterfactual regret minimization (vr-mccfr) for extensive form games using baselines. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[95] Richard Gibson, Marc Lanctot, Neil Burch, Duane Szafron, and Michael Bowling. Generalized sampling and variance in counterfactual regret minimization. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, pages 1355–1361, 2012.

[96] Kevin Waugh, Dustin Morrill, J. Andrew Bagnell, and Michael Bowling. Solving games with functional regret estimation. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2138–2144, 2015.

[97] Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *Proceedings of the 35th International Conference on International Conference on Machine Learning*, 2018.

[98] Noam Brown and Tuomas Sandholm. Solving imperfect-information games via discounted regret minimization, 2018.

[99] Eric Steinberger. Single deep counterfactual regret minimization, 2019.

[100] Peter Jin, Kurt Keutzer, and Sergey Levine. Regret minimization for partially observable deep reinforcement learning. volume 80 of *Proceedings of Machine Learning Research*. PMLR, 2017.

[101] Hui Li, Kailiang Hu, Zhibang Ge, Tao Jiang, Yuan Qi, and Le Song. Double neural counterfactual regret minimization. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[102] Noam Brown and Tuomas Sandholm. Libratus: The superhuman ai for no-limit poker. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, page 5226–5228, 2017.

[103] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[104] Katja Hofmann Ian A. Kash, Michael Sullins. Combining no-regret and q-learning, 2019.

[105] Yoav Shoham, Rob Powers, and Trond Grenager. If multi-agent learning is the answer, what is the question? *Artif. Intell.*, 171(7):365–377, May 2007.

[106] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E. Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, Oct 2019.