# Decentralized Mean Field Games

**Sriram Ganapathi Subramanian**[1, 3], **Matthew E. Taylor**[2, 4], **Mark Crowley**[1], **Pascal Poupart**[1, 3]

[1] University of Waterloo, Waterloo, Ontario, Canada
[2] University of Alberta, Edmonton, Alberta, Canada
[3] Vector Institute, Toronto, Ontario, Canada
[4] Alberta Machine Intelligence Institute (Amii), Edmonton, Alberta, Canada
s2ganapa@uwaterloo.ca, matthew.e.taylor@ualberta.ca, mcrowley@uwaterloo.ca, ppoupart@uwaterloo.ca

## Abstract

Multiagent reinforcement learning algorithms have not been widely adopted in large scale environments with many agents as they often scale poorly with the number of agents. Using mean field theory to aggregate agents has been proposed as a solution to this problem. However, almost all previous methods in this area make a strong assumption of a centralized system where all the agents in the environment learn the same policy and are effectively indistinguishable from each other. In this paper, we relax this assumption about indistinguishable agents and propose a new mean field system known as *Decentralized Mean Field Games*, where each agent can be quite different from others. All agents learn independent policies in a decentralized fashion, based on their local observations. We define a theoretical solution concept for this system and provide a fixed point guarantee for a $Q$-learning based algorithm in this system. A practical consequence of our approach is that we can address a 'chicken-and-egg' problem in empirical mean field reinforcement learning algorithms. Further, we provide $Q$-learning and actor-critic algorithms that use the decentralized mean field learning approach and give stronger performances compared to common baselines in this area. In our setting, agents do not need to be clones of each other and learn in a fully decentralized fashion. Hence, for the first time, we show the application of mean field learning methods in fully competitive environments, large-scale continuous action space environments, and other environments with heterogeneous agents. Importantly, we also apply the mean field method in a ride-sharing problem using a real-world dataset. We propose a decentralized solution to this problem, which is more practical than existing centralized training methods.

## Introduction

Most multiagent reinforcement learning (MARL) algorithms are not tractable when applied to environments with many agents (infinite in the limit) as these algorithms are exponential in the number of agents (Busoniu, Babuska, and De Schutter 2006). One exception is a class of algorithms that use the mean field theory (Stanley 1971) to approximate the many agent setting to a two agent setting, where the second agent is a mean field distribution of all agents representing the average effect of the population. This makes MARL algorithms tractable since, effectively, only two agents are being modelled. Lasry and Lions (2007) introduced the framework of a *mean field game* (MFG), which incorporates the mean field theory in MARL. In MARL, the mean field can be a population state distribution (Huang 2010) or action distribution (Yang et al. 2018) of all the other agents in the environment.

MFGs have three common assumptions. First, each agent does not have access to the local information of the other agents. However, it has access to accurate global information regarding the mean field of the population. Second, all agents in the environment are independent, homogeneous, and indistinguishable. Third, all agents maintain interactions with others only through the mean field. These assumptions (especially the first two) severely restrict the potential of using mean field methods in real-world environments. The first assumption is impractical, while the second assumption implies that all agents share the same state space, action space, reward function, and have the same objectives. Further, given these assumptions, prior works use centralized learning methods, where all agents learn and update a shared centralized policy. These two assumptions are only applicable to cooperative environments with extremely similar agents. Theoretically, the agent indices are omitted since all agents are interchangeable (Lasry and Lions 2007). We will relax the first two assumptions. In our case, the agents are not interchangeable and can each formulate their own policies during learning that differs from others. Also, we will not assume the availability of the immediate global mean field. Instead, agents only have local information and use modelling techniques (similar to opponent modelling common in MARL (Hernandez-Leal, Kartal, and Taylor 2019)) to effectively model the mean field during the training process. We retain the assumption that each agent's impact on the environment is infinitesimal (Huang 2010), and hence agents calculate best responses only to the mean field. Formulating best responses to each individual agent is intractable and unnecessary (Lasry and Lions 2007).

The solution concepts proposed by previous mean field methods have been either the centralized Nash equilibrium (Yang et al. 2018) or a closely related mean field equilibrium (Lasry and Lions 2007). These solution concepts are centralized as they require knowledge of the current policy of all other agents or other global information, even in non-cooperative environments. Verifying their existence is infeasible in many practical environments (Neumann 1928).

This work presents a new kind of mean field system, *Decentralized Mean Field Games* (DMFGs), which uses a decentralized information structure. This new formulation of the mean field system relaxes the assumption of agents' indistinguishability and makes mean field methods applicable to numerous real-world settings. Subsequently, we provide a decentralized solution concept for learning in DMFGs, which will be more practical than the centralized solution concepts considered previously. We also provide a fixed point guarantee for a $Q$-learning based algorithm in this system.

A 'chicken-and-egg' problem exists in empirical mean field reinforcement learning algorithms where the mean field requires agent policies, yet the policies cannot be learned without the global mean field (Yang and Wang 2020). We show that our formulation can address this problem, and we provide practical algorithms to learn in DMFGs. We test our algorithms in different types of many agent environments. We also provide an example of a ride-sharing application that simulates demand and supply based on a real-world dataset.

## Background

**Stochastic Game**: An $N$-player stochastic (Markovian) game can be represented as a tuple $\langle \mathcal{S}, \mathcal{A}^1, \dots, \mathcal{A}^N, r^1, \dots, r^N, p, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}^j$ represents the action space of the agent $j \in \{1, \dots, N\}$, and $r^j : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \to \mathcal{R}$ represents the reward function of $j$. Also, $p : \mathcal{S} \times \mathcal{A}^1 \times \dots \times \mathcal{A}^N \to \Omega(\mathcal{S})$ represents the transition probability that determines the next state given the current state and the joint action of all agents. Here $\Omega(\mathcal{S})$ is a probability distribution over the state space. In the stochastic game, agents aim to maximize the expected discounted sum of rewards, with discount factor $\gamma \in [0, 1)$.

Each agent $j$ in the stochastic game formulates a policy, which is denoted by $\pi^j : \mathcal{S} \to \Omega(\mathcal{A}^j)$ where the joint policy is represented as $\boldsymbol{\pi} = [\pi^1, \dots, \pi^N]$ for all $s \in \mathcal{S}$. Given an initial state $s$, the value function of the agent $j$ under the joint policy $\boldsymbol{\pi}$ can be represented as $v^j(s|\boldsymbol{\pi}) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}[r_t^j | s_0 = s, \boldsymbol{\pi}]$. The solution concept is the *Nash equilibrium* (NE) (Hu and Wellman 2003). This is represented by a joint policy $\boldsymbol{\pi}_* = [\pi_*^1, \dots, \pi_*^N]$, such that, for all $\pi^j$, $v^j(s|\pi_*^j, \boldsymbol{\pi}_*^{-j}) \geq v^j(s|\pi^j, \boldsymbol{\pi}_*^{-j})$, for all agents $j \in \{1, \dots, N\}$ and all states $s \in \mathcal{S}$. The notation $\boldsymbol{\pi}_*^{-j}$ represents the joint policy of all agents except the agent $j$.

**Mean Field Game**: MFG was introduced as a framework to solve the stochastic game when the number of agents $N$ is very large (Lasry and Lions 2007; Huang et al. 2006). In this setting, calculating the best response to each individual opponent is intractable, so each agent responds to the aggregated state distribution $\mu_t \triangleq \lim_{N \to \infty} \frac{\sum_{j=1, j \neq i} \mathbf{1}(s_t^j)}{N}$, known as the mean field. Let $\boldsymbol{\mu} \triangleq \{\mu_t\}_{t=0}^{\infty}$. MFG assumes that all agents are identical (homogeneous), indistinguishable, and interchangeable (Lasry and Lions 2007). Given this assumption, the environment changes to a single-agent stochastic control problem where all agents share the same policy (Saldi, Basar, and Raginsky 2018). Hence, $\pi^1 = \dots = \pi^N = \boldsymbol{\pi}$. The theoretical formulation focuses on a representative player, and the solution of this player (optimal policy) obtains the solution for the entire system. The value function of a representative agent can be given as $V(s, \boldsymbol{\pi}, \boldsymbol{\mu}) \triangleq \mathbb{E}_{\boldsymbol{\pi}}\big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t, \mu_t) | s_0 = s\big]$, where $s$ and $a$ are the state and action of the representative agent, respectively. The transition dynamics is represented as $P(s_t, a_t, \mu_t)$, where the dependence is on the mean field distribution. All agents share the same reward function $r(s_t, a_t, \mu_t)$. The action comes from the policy $\pi_t(a_t|s_t, \mu_t)$.

The solution concept for the MFG is the mean field equilibrium (MFE). A tuple $(\boldsymbol{\pi}_{MFG}^*, \boldsymbol{\mu}_{MFG}^*)$ is a mean field equilibrium if, for any policy $\boldsymbol{\pi}$, an initial state $s \in \mathcal{S}$ and given mean field $\boldsymbol{\mu}_{MFG}^*$, $V(s, \boldsymbol{\pi}_{MFG}^*, \boldsymbol{\mu}_{MFG}^*) \geq V(s, \boldsymbol{\pi}, \boldsymbol{\mu}_{MFG}^*)$. Additionally, $\boldsymbol{\mu}_{MFG}^*$ is the mean field obtained when all agents play the same $\boldsymbol{\pi}_{MFG}^*$ at each $s \in \mathcal{S}$.

**Mean Field Reinforcement Learning** (MFRL): MFRL, introduced by Yang et al. (2018), is another approach for learning in stochastic games with a large number of agents. Here, the empirical mean action is used as the mean field, which each agent then uses to update its $Q$-function and Boltzmann policy. Each agent is assumed to have access to the global state, and it takes local action from this state. In MFRL, the $Q$-function of the agent $j$ is updated as,

$$\begin{aligned} &Q^j(s_t, a_t^j, \mu_t^a) \\ &\quad = (1 - \alpha) Q^j(s_t, a_t^j, \mu_t^a) + \alpha[r_t^j + \gamma v^j(s_{t+1})] \end{aligned} \tag{1}$$

where

$$v^j(s_{t+1}) = \sum_{a_{t+1}^j} \pi^j(a_{t+1}^j | s_{t+1}, \mu_t^a) Q^j(s_{t+1}, a_{t+1}^j, \mu_t^a) \tag{2}$$

$$\mu_t^a = \frac{1}{\mathcal{N}} \sum_j a_t^j, a_t^j \sim \pi^j(\cdot | s_t, \mu_{t-1}^a) \tag{3}$$

$$\pi^j(a_t^j | s_t, \mu_{t-1}^a) = \frac{\exp(-\hat{\beta} Q^j(s_t, a_t^j, \mu_{t-1}^a)}{\sum_{a_t^{j'} \in A^j} \exp(-\hat{\beta} Q^j(s_t, a_t^{j'}, \mu_{t-1}^a))} \tag{4}$$

where $s_t$ is the global old state, $s_{t+1}$ is the global resulting state, $r_t^j$ is the reward of $j$ at time $t$, $v^j$ is the value function of $j$, $\mathcal{N}$ is the total number of agents, and $\hat{\beta}$ represents the Boltzmann parameter. The action $a^j$ is assumed to be discrete and represented using one-hot encoding. Like stochastic games, MFRL uses the NE as the solution concept. The global mean field $\mu_t^a$ captures the action distribution of all agents. To address this global limitation, Yang et al. (2018) specify mean field calculation over certain neighbourhoods for each agent. However, an update using Eq. 3 requires each agent to have access to all other agent policies or the global mean field to use the centralized concept (NE). We omit an expectation in Eq. 2 since Yang et al. (2018) guaranteed that their updates will be greedy in the limit with infinite exploration (GLIE).

From Eq. 3 and Eq. 4, it can be seen that the current action depends on the mean field and the mean field depends on the current action. To resolve this 'chicken-and-egg' problem, Yang et al. (2018) simply use the previous mean field action to decide the current action, as in Eq. 4. This can lead to a loss of performance since the agents are formulating best responses to the previous mean field action $\mu_{t-1}^a$, while they are expected to respond to the current mean field action $\mu_t^a$.

## Related Work

MFGs were first proposed in Huang, Caines, and Malhamé (2003), while a comprehensive development of the system and principled application methods were given later in Lasry and Lions (2007). Subsequently, learning algorithms were proposed for this framework. Subramanian and Mahajan (2019) introduce a restrictive form of MFG (known as stationary MFG) and provide a model-free policy-gradient (Sutton et al. 1999; Konda and Tsitsiklis 1999) algorithm along with convergence guarantees to a local Nash equilibrium. On similar lines, Guo et al. (2019) provide a model-free $Q$-learning algorithm (Watkins and Dayan 1992) for solving MFGs, also in the stationary setting. The assumptions in these works are difficult to verify in real-world environments. Particularly, Guo et al. (2019) assume the presence of a game engine (simulator) that accurately provides mean field information to all agents at each time step, which is not practical in many environments. Further, other works depend on fictitious play updates for the mean field parameters (Hadikhanloo and Silva 2019; Elie et al. 2020), which involves the strong assumption that opponents play stationary strategies. All these papers use the centralized setting for the theory and the experiments.

Prior works (Gomes, Mohr, and Souza 2010; Adlakha, Johari, and Weintraub 2015; Saldi, Basar, and Raginsky 2018) have established the existence of a (centralized) mean field equilibrium in the discrete-time MFG under a discounted cost criterion, in finite and infinite-horizon settings. Authors have also studied the behaviour of iterative algorithms and provided theoretical analysis for learning of the non-stationary (centralized) mean field equilibrium in infinite-horizon settings (Więcek and Altman 2015; Więcek 2020; Anahtarci, Kariksiz, and Saldi 2019). We provide similar guarantees in the decentralized setting with possibly heterogeneous agents.

Yang et al. (2018) introduces MFRL that uses a mean field approximation through the empirical mean action, and provides two practical algorithms that show good performance in large MARL settings. The approach is model-free and the algorithms do not need strong assumptions regarding the nature of the environment. However, they assume access to global information that needs a centralized setting for both theory and experiments. MFRL has been extended to multiple types (Subramanian et al. 2020) and partially observable environments (Subramanian et al. 2021). However, unlike us, Subramanian et al. (2020) assumes that agents can be divided into a finite set of types, where agents within a type are homogeneous. Subramanian et al. (2021) relaxes the assumption of global information in MFRL, however, it still uses the Nash equilibrium as the solution concept. Further, that work contains some assumptions regarding the existence of conjugate priors, which is hard to verify in real-world environments. Additional related work is provided in Appendix L.

## Decentralized Mean Field Game

The DMFG model is specified by $\langle \boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{A}}, p, \boldsymbol{R}, \mu_0 \rangle$, where $\boldsymbol{\mathcal{S}} = \mathcal{S}^1 \times \cdots \times \mathcal{S}^N$ represents the state space and $\boldsymbol{\mathcal{A}} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^N$ represents the joint action space. Here, $\mathcal{S}^j$ represents the state space of an agent $j \in \{1, \ldots, N\}$ and $\mathcal{A}^j$ represents the action space of $j$. As in MFGs, we are considering the infinite population limit of the game, where the set of agents $N$, satisfy $N \to \infty$. Similar to the MFG formulation in several prior works (Lasry and Lions 2007; Huang et al. 2006; Saldi, Basar, and Raginsky 2018), we will specify that both the state and action spaces are Polish spaces. Particularly, $\mathcal{A}^j$ for all agents $j$, is a compact subset of a finite dimensional Euclidean space $\Re^d$ with the Euclidean distance norm $|| \cdot ||$. Since all agents share the same environment, for simplicity, we will also assume that the state spaces of all the agents are the same $\mathcal{S}^1 = \cdots = \mathcal{S}^N = \mathcal{S}$ and are locally compact. Since the state space is a complete separable metric space (Polish space), it is endowed with a metric $d_X$. The transition function $p : \mathcal{S} \times \mathcal{A}^j \times \mathcal{P}(\mathcal{S}) \to \mathcal{P}(\mathcal{S})$ determines the next state of any $j$ given the current state and action of $j$, and the probability distribution of the state in the system (represented by the mean field). The reward function is represented as a set $\boldsymbol{R} = \{R^1, \ldots, R^N\}$, where, $R^j : \mathcal{S} \times \mathcal{A}^j \times \mathcal{P}(\mathcal{S}) \to [0, \infty)$ is the reward function of $j$.

Recall that a DMFG has two major differences as compared to MFG and MFRL. 1) DMFG does not assume that the agents are indistinguishable and homogeneous (agent indices are retained). 2) DMFG does not assume that each agent can access the global mean field of the system. However, each agents' impact on the environment is infinitesimal, and therefore all agents formulate best responses only to the mean field of the system (no per-agent modelling is required).

As specified, in DMFG, the transition and reward functions for each agent depends on the mean field of the environment, represented by $\boldsymbol{\mu} \triangleq (\mu_t)_{t \geq 0}$, with the initial mean field represented as $\mu_0$. For DMFG, mean field can either correspond to the state distribution $\mu_t \triangleq \lim_{N \to \infty} \frac{\sum_{j=1}^{N} \mathbf{1}(s_t^j)}{N}$ or the action distribution $\mu_t^a$ as in Eq. 3. Without loss of generality we use the mean field as the state distribution $\mu_t$ (represented by $\mathcal{P}(\mathcal{S})$), as done in prior works (Lasry and Lions 2007; Elliott, Li, and Ni 2013). However, our setting and theoretical results will hold for the mean field as action distribution $\mu_t^a$ as well.

In the DMFG, each agent $j$ will not have access to the true mean field of the system and instead use appropriate techniques to actively model the mean field through exploration. The agent $j$ holds an estimate of the actual mean field represented by $\mu_t^j$. Let $\boldsymbol{\mu}^j \triangleq (\mu_t^j)_{t \geq 0}$. Let, $\mathcal{M}$ be used to denote the set of mean fields $\{\boldsymbol{\mu}^j \in \mathcal{P}(\mathcal{S})\}$. A Markov policy for an agent $j$ is a stochastic kernel on the action space $\mathcal{A}^j$ given the immediate local state ($s_t^j$) and the agent's current estimated mean field $\mu_t^j$, i.e. $\pi_t^j : \mathcal{S} \times \mathcal{P}(\mathcal{S}) \to \mathcal{P}(\mathcal{A}^j)$, for each $t \geq 0$. Alternatively, a non-Markov policy will depend on the entire state-action history of game play. We will use $\Pi^j$ to denote a set of all policies (both Markov and non-Markov) for the agent $j$. Let $s_t^j$ represent the state of an agent $j$ at time $t$ and $a_t^j$ represent the action of $j$ at $t$. Then an agent $j$ tries to maximize the objective function given by the following equation (where $r^j$ denotes the immediate reward obtained by the agent $j$ and $\beta \in [0, 1)$ denotes the discount factor),

$$J_{\boldsymbol{\mu}}^j(\pi^j) \triangleq \mathbb{E}^{\pi^j}[\textstyle\sum_{t=0}^{\infty} \beta^t r^j(s_t^j, a_t^j, \mu_t)]. \qquad (5)$$

In line with prior works (Yang et al. 2018; Subramanian et al. 2020), we assume that each agent's sphere of influence is restricted by its neighbourhood, where it conducts exploration. Using this assumption, we assert that, after a finite $t$, the mean field estimate will accurately reflect the true mean field for $j$, in its neighbourhood denoted as $\mathcal{N}^j$. This assumption specifies that agents have full information in their neighbourhood, and they can use modelling techniques to obtain accurate mean field information within the neighbourhood (also refer to flocking from Perrin et al. (2021)).

**Assumption 1.** *There exists a finite time $T$ and a neighbourhood $\mathcal{N}^j$, such that for all $t > T$, the mean field estimate of an agent $j \in 1, \dots, N$ satisfies $(\forall s^j \in \mathcal{N}^j)$ $\boldsymbol{\mu}^j(s^j) = \boldsymbol{\mu}(s^j)$. Also, $\forall s^j \in \mathcal{N}^j$, we have, $p(\cdot|s^j_t, a^j_t, \mu^j_t) = p(\cdot|s^j_t, a^j, \mu_t)$ and $r^j(\cdot|s^j_t, a^j_t, \mu^j_t) = r^j(\cdot|s^j_t, a^j_t, \mu_t)$.*

Let us define a set $\Phi : \mathcal{M} \to 2^{\Pi}$ as $\Phi(\boldsymbol{\mu^j}) = \{\pi^j \in \Pi^j : \pi^j \text{ is optimal for } \boldsymbol{\mu^j}\}$. Conversely, for $j$, we define a mapping $\Psi : \boldsymbol{\Pi} \to \mathcal{M}$ as, given a policy $\pi^j \in \Pi^j$, the mean field state estimate $\boldsymbol{\mu}^j \triangleq \Psi(\pi^j)$ can be constructed as,

$$\mu^j_{t+1}(\cdot) = \int_{\mathcal{S}\times\mathcal{A}^j} p(\cdot|s^j_t, a^j_t, \mu_t)\mathcal{P}^{\pi^j}(a^j_t|s^j_t, \mu^j_t)\mu^j_t(s^j_t). \tag{6}$$

Here $\mathcal{P}^{\pi^j}$ is a probability measure induced by $\pi^j$. Later (in Theorem 1) we will prove that restricting ourselves to Markov policies is sufficient in a DMFG, and hence $\mathcal{P}^{\pi^j} = \pi^j$.

Now, we can define the *decentralized mean field equilibrium* (DMFE) which is the solution concept for this game.

**Definition 1.** *The decentralized mean field equilibrium of an agent $j$ is represented as a pair $(\pi^j_*, \mu^j_*) \in \Pi^j \times \mathcal{M}$ if $\pi^j_* \in \Phi(\mu^j_*)$ and $\mu^j_* = \Psi(\pi^j_*)$. Here $\pi^j_*$ is the best response to $\mu^j_*$ and $\mu^j_*$ is the mean field estimate of $j$ when it plays $\pi^j_*$.*

The important distinction between DMFE and centralized concepts, such as NE and MFE, is that DMFE does not rely on the policy information of other agents. MFE requires all agents to play the same policy, and NE requires all agents to have access to other agents' policies. DMFE has no such constraints. Hence, this decentralized solution concept is more practical than NE and MFE. In Appendix F, we summarize the major differences between the DMFG, MFG and MFRL.

## Theoretical Results

We provide a set of theorems that will first guarantee the existence of the DMFE in a DMFG. Further, we will show that a simple $Q$-learning update will converge to a fixed point representing the DMFE. We will borrow relevant results from prior works in centralized MFGs in our theoretical guarantees. Particularly, we aim to adapt the results and proof techniques in works by Saldi, Basar, and Raginsky (2018), Lasry and Lions (2007), and Anahtarci, Kariksiz, and Saldi (2019) to the decentralized setting. The statements of all our theorems are given here, while the proofs are in Appendices A – E.

Similar to an existing result from centralized MFG (Lasry and Lions 2007), in the DMFG, restricting policies to only Markov policies would not lead to any loss of optimality. We use $\Pi^j_M$ to denote the set of Markov policies for the agent $j$.

**Theorem 1.** *For any mean field, $\boldsymbol{\mu} \in \mathcal{M}$, and an agent $j \in \{1, \dots, N\}$, we have,*

$$\sup_{\pi^j \in \Pi^j} J^j_{\boldsymbol{\mu}}(\pi^j) = \sup_{\pi^j \in \Pi^j_M} J^j_{\boldsymbol{\mu}}(\pi^j). \tag{7}$$

Next, we show the existence of a DMFE under a set of assumptions similar to those previously used in the centralized MFG (Lasry and Lions 2007; Saldi, Basar, and Raginsky 2018). The assumptions pertain to bounding the reward function and imposing restrictions on the nature of the mean field (formal statements in Appendix B). We do not need stronger assumptions than those previously considered for the MFGs.

**Theorem 2.** *An agent $j \in \{1, \dots, N\}$ in the DMFG admits a decentralized mean field equilibrium $(\pi^j_*, \mu^j_*) \in \Pi^j \times \mathcal{M}$.*

We use $\mathcal{C}$ to denote a set containing bounded functions in $\mathcal{S}$. Now, we define a decentralized mean field operator ($H$),

$$\begin{aligned} H : \mathcal{C} \times \mathcal{P}(\mathcal{S}) \ni (Q^j, \boldsymbol{\mu}^j) & \\ \to (H_1(Q^j, \boldsymbol{\mu}^j), H_2(Q^j, \boldsymbol{\mu}^j)) \in \mathcal{C} \times \mathcal{P}(\mathcal{S}) & \end{aligned} \tag{8}$$

where

$$\begin{aligned} H_1(Q^j, \boldsymbol{\mu}^j)(s^j_t, a^j_t) &\triangleq r^j(s^j_t, a^j_t, \mu_t) \\ &+ \beta \int_{\mathcal{S}} Q^j_{\max_{a^j}}(s^j_{t+1}, a^j, \mu^j_{t+1}) p(s^j_{t+1}|s^j_t, a^j_t, \mu_t) \end{aligned} \tag{9}$$

$$\begin{aligned} H_2(Q^j, \boldsymbol{\mu}^j)(\cdot) & \\ \triangleq \int_{\mathcal{S}\times\mathcal{A}^j} p\Big(\cdot\,|s^j_t, \pi^j(s^j_t, Q^j_t, \mu^j_t), \mu_t\Big)\mu^j_t(s) & \end{aligned} \tag{10}$$

for an agent $j$. Here, $\pi^j$ is a maximiser of the operator $H_1$.

For the rest of the theoretical results, we consider a set of assumptions different from those needed for Theorem 2. Here we assume that the reward and transition functions are Lipschitz continuous with a suitable Lipschitz constant. The Lipschitz continuity assumption is quite common in the mean field literature (Yang et al. 2018; Lasry and Lions 2007; Subramanian et al. 2020). We also consider some further assumptions regarding the nature of gradients of the value function (formal statements in Appendix C). All assumptions are similar to those considered before for the analysis in the centralized MFGs (Lasry and Lions 2007; Anahtarci, Kariksiz, and Saldi 2019; Huang 2010). First, we provide a theorem regarding the nature of operator $H$. Then, we provide another theorem showing that $H$ is a contraction.

**Theorem 3.** *The decentralized mean field operator $H$ is well-defined, i.e., this operator maps $\mathcal{C} \times \mathcal{P}(\mathcal{S})$ to itself.*

**Theorem 4.** *Let $\mathcal{B}$ represent the space of bounded functions in $\mathcal{S}$. Then the mapping $H : \mathcal{C} \times \mathcal{P}(\mathcal{S}) \to \mathcal{C} \times \mathcal{P}(\mathcal{S})$ is a contraction in the norm of $\mathcal{B}(\mathcal{S})$.*

Since $H$ is a contraction, by the Banach fixed point theorem (Shukla, Balasubramanian, and Pavlović 2016), we can obtain that fixed point using the $Q$ iteration algorithm given by Alg. 1. We provide this result in the next theorem.

**Theorem 5.** *Let the $Q$-updates in Algorithm 1 converge to $(Q^j_*, \mu^j_*)$ for an agent $j \in \{1, \dots, N\}$. Then, we can construct a policy $\pi^j_*$ from $Q^j_*$ using the relation,*

$$\pi^j_*(s^j) = \arg\max_{a^j \in \mathcal{A}^j} Q^j_*(s^j, a^j, \mu^j_*). \tag{11}$$

*Then the pair $(\pi^j_*, \mu^j_*)$ is a DMFE.*

Hence, from the above results, we have proved that a DMFG admits a DMFE, and an iterative algorithm using $Q$-updates can arrive at this equilibrium. This provides a fixed point for Alg. 1, to which the $Q$-values converge.

Algorithm 1: Q-learning for DMFG

```
1: For each agent $j \in \{1, \ldots, N\}$, start with initial $Q$-function $Q_0^j$ and the initial mean field state estimate $\mu_0^j$
2: while $(Q_n^j, \mu_n^j) \neq (Q_{n-1}^j, \mu_{n-1}^j)$ do
3:     $(Q_{n+1}^j, \mu_{n+1}^j) = H(Q_n^j, \mu_n^j)$
4: end while
5: Return the fixed point $(Q_*^j, \mu_*^j)$ of $H$
```

## Algorithms

We will apply the idea of decentralized updates to the model-free MFRL framework. We modify the update equations in MFRL and make them decentralized, where agents only observe their local state and do not have access to the immediate global mean field. Our new updates are:

$$Q^j(s_t^j, a_t^j, \mu_t^{j,a}) = (1-\alpha)Q^j(s_t^j, a_t^j, \mu_t^{j,a}) + \alpha[r_t^j + \gamma v^j(s_{t+1}^j)] \tag{12}$$

where

$$v^j(s_{t+1}^j) = \sum_{a_{t+1}^j} \pi^j(a_{t+1}^j | s_{t+1}^j, \mu_{t+1}^{j,a}) Q^j(s_{t+1}^j, a_{t+1}^j, \mu_{t+1}^{j,a}) \tag{13}$$

$$\mu_t^{j,a} = f^j(s_t^j, \hat{\mu}_{t-1}^{j,a}) \tag{14}$$

$$\text{and } \pi^j(a_t^j | s_t^j, \mu_t^{j,a}) = \frac{\exp(-\hat{\beta} Q^j(s_t^j, a_t^j, \mu_t^{j,a}))}{\sum_{a_t^{j'} \in A^j} \exp(-\hat{\beta} Q^j(s_t^j, a_t^{j'}, \mu_t^{j,a}))} \tag{15}$$

Here, $s_t^j$ is the local state and $\mu_t^{j,a}$ is the mean field action estimate for the agent $j$ at time $t$ and $\hat{\mu}_{t-1}^{j,a}$ is the observed local mean field action of $j$ at $t-1$. Other variables have the same meaning as Eq. 1 – Eq. 4. In Eq. 14, the mean field estimate for $j$ is updated using a function of the current state and the previous local mean field. Opponent modelling techniques commonly used in MARL (Hernandez-Leal, Kartal, and Taylor 2019) can be used here. We use the technique of He and Boyd-Graber (2016), that used a neural network to model the opponent agent(s). In our case, we use a fully connected neural network (2 Relu layers of 50 nodes and an output softmax layer) to model the mean field action. The network takes the current state and previous mean field action as inputs and outputs the estimated current mean field. This network is trained using a mean square error between the estimated mean field action from the network (Eq. 14) and the observed local mean field (local observation of actions of other agents $\hat{\mu}_t^{j,a}$) after action execution. The policy in Eq. 15 does not suffer from the 'chicken-and-egg' problem of Eq. 4 since it depends on the current mean field estimate, unlike MFRL which used the previous global mean field in Eq. 4.

We provide a neural network-based $Q$-learning implementation for our update equations, namely *Decentralized Mean Field Game Q-learning* (DMFG-QL), and an actor-critic implementation, *Decentralized Mean Field Game Actor-Critic* (DMFG-AC). Detailed description of the algorithms are in Appendix H (see Algs. 2 and 3). A complexity analysis is in Appendix K, and hyperparameter details are in Appendix J.

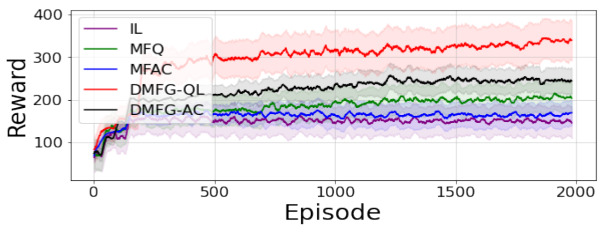
## Experiments and Results

In this section we study the performance of our algorithms. The code for experiments has been open-sourced (Subramanian 2021). We provide the important elements of our domains here, while the complete details are in Appendix I.

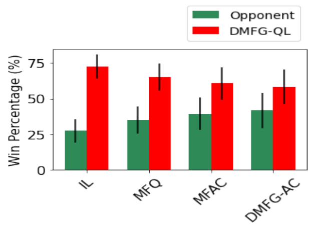
The first five domains belong to the MAgent environment (Zheng et al. 2018). We run the experiments in two phases, training and execution. Analogous to experiments conducted in previous mean field studies (Yang et al. 2018; Subramanian et al. 2020), all agents train against other agents playing the same algorithm for 2000 games. This is similar to multiagent training using self-play (Shoham, Powers, and Grenager 2003). The trained agents then enter into an execution phase, where the trained policies are simply executed. The execution is run for 100 games, where algorithms may compete against each other. We consider three baselines, independent $Q$-learning (IL) (Tan 1993), mean field $Q$-learning (MFQ) (Yang et al. 2018), and mean field actor-critic (MFAC) (Yang et al. 2018). Each agent in our implementations learns in a decentralized fashion, where it maintains its own networks and learns from local experiences. This is unlike centralized training in prior works (Yang et al. 2018; Guo et al. 2019). We repeat experiments 30 times, and report the mean and standard deviation. Wall clock times are given in Appendix M.

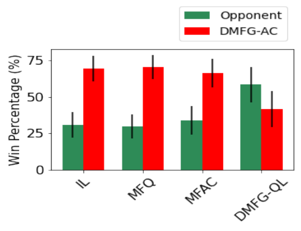
First, we consider the mixed cooperative-competitive Battle game (Zheng et al. 2018). This domain consists of two teams of 25 agents each. Each agent is expected to cooperate with agents within the team and compete against agents of the other team to win the battle. For the training phase, we plot the cumulative rewards per episode obtained by the agents of the first team for each algorithm in Fig. 1(a). The performance of the second team is also similar (our environment is not zero-sum). From the results, we see that DMFG-QL performs best while the others fall into a local optimum and do not get the high rewards. The DMFG-AC algorithm comes second. It has been noted previously (Yang et al. 2018) that $Q$-learning algorithms often perform better compared to their actor-critic counterparts in mean field environments. MFQ and MFAC (using the previous mean field information) performs poorly compared to DMFG-QL and DMFG-AC (using the current estimates). Finally, IL loses out to others due to its independent nature. In execution, one team trained using one algorithm competes against another team from a different algorithm. We plot the percentage of games won by each algorithm in a competition against DMFG-QL and DMFG-AC. A game is won by the team that kills more of its opponents. The performances are in Fig. 1(b) and (c), where DMFG-QL performs best. In Appendix G, we show that DFMG-QL can accurately model the true mean field in the Battle game.

The second domain is the heterogeneous Combined Arms environment. This domain is a mixed setting similar to Battle except that each team consists of two different types of agents, ranged and melee, with distinct action spaces. Each team has 15 ranged and 10 melee agents. This environment is different from those considered in Subramanian et al. (2020), which formulated each team as a distinct type, where agents within a team are homogeneous. The ranged agents are faster and attack further, but can be killed quickly. The melee agents are slower but are harder to kill. We leave out MFQ and MFAC



(a) Training

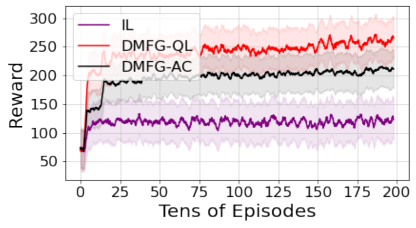


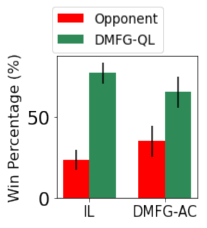


(b) Execution vs. DMFG-QL (c) Execution vs. DMFG-AC

Figure 1: Battle results. In (a) solid lines show average and shaded regions represent standard deviation. In (b) and (c) bars are average and black lines represent standard deviation.

for this experiment, since both these algorithms require the presence of fully homogeneous agents. The experimental procedure is the same as in Battle. From the results we see that DMFG-QL performs best in both phases (see Fig. 2).

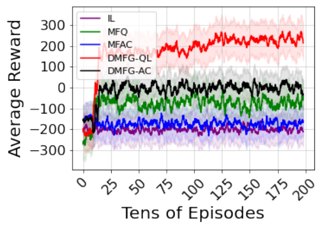


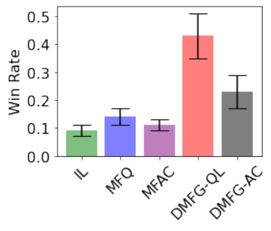


(a) Training (b) Execution

Figure 2: Combined Arms results

Next is the fully competitive Gather environment. This contains 30 agents trying to capture limited food. All the agents compete against each other for capturing food and could resort to killing others when the food becomes scarce. We plot the average rewards obtained by each of the five algorithms in the training phase (Fig. 3(a)). DMFG-QL once again obtains the maximum performance. In competitive environments, actively formulating the best responses to the current strategies of opponents is crucial for good performances. Predictably, the MFQ and MFAC algorithms (relying on previous information) lose out. For execution, we sample (at random) six agents from each of the five algorithms to make a total of 30. We plot the percentage of games won by each algorithm in a total of 100 games. A game is determined to have been won by the agent obtaining the most rewards. Again, DMFG-QL shows the best performance during execution (Fig. 3(b)).

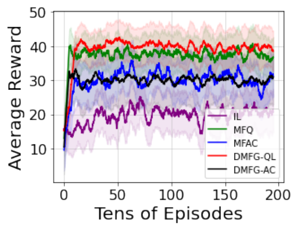


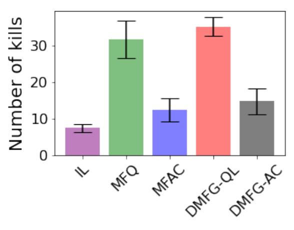


(a) Training (b) Execution

Figure 3: Gather results

The next domain is a fully cooperative Tiger-Deer environment. In this environment, a team of tigers aims to kill deer. The deer are assumed to be part of the environment moving randomly, while the tigers are agents that learn to coordinate with each other to kill the deer. At least two tigers need to attack a deer in unison to gain large rewards. Our environment has 20 tigers and 101 deer. In the training phase, we plot the average reward obtained by the tigers (Fig. 4(a)). The performance of MFQ almost matches that of DMFG-QL and the performance of DMFG-AC matches MFAC. In a cooperative environment, best responses to actively changing strategies of other agents are not as critical as in competitive environments. Here all agents aid each other and using the previous time information (as done in MFQ and MFAC) does not hurt performance as much. For execution, a set of 20 tigers from each algorithm execute their policy for 100 games. We plot the average number of deer killed by the tigers for each algorithm. DMFG-QL gives the best performance (Fig. 4(b)).

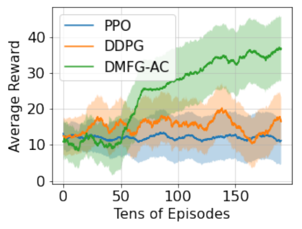




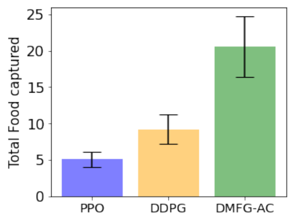
(a) Training (b) Execution

Figure 4: Tiger-Deer results

The next domain is the continuous action Waterworld domain, first introduced by Gupta, Egorov, and Kochenderfer (2017). This is also a fully cooperative domain similar to Tiger-Deer, where a group of 25 pursuer agents aim to capture a set of food in the environment while actively avoiding poison. The action space corresponds to a continuous thrust variable. We implement DMFG-AC in this domain, where the mean field is a mixture of Dirac deltas of actions taken by all agents. The experimental procedure is the same as Tiger-Deer. For this continuous action space environment, we use proximal policy optimization (PPO) (Schulman et al. 2017) and deep deterministic policy gradient (DDPG) (Lillicrap et al. 2016), as baselines. We see that DMFG-AC obtains the



(a) Training



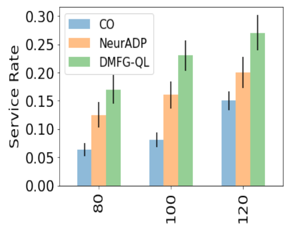
(b) Execution

Figure 5: Waterworld results

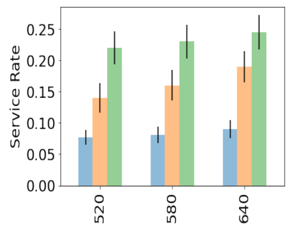
best performance in both phases (refer to Fig. 5(a) and (b)).

Our final environment is a real-world *Ride-pool Matching Problem* (RMP) introduced by Alonso-Mora et al. (2017). This problem pertains to improving the efficiency of vehicles satisfying ride requests as part of ride-sharing platforms such as UberPool. In our environment, ride requests come from the open source New York Yellow Taxi dataset (NYYellowTaxi 2016). The road network (represented as a grid with a finite set of nodes or road intersections) contains a simulated set of vehicles (agents) that aim to serve the user requests. Further details about this domain are in Appendix I. We consider two baselines in this environment. The first is the method from Alonso-Mora et al. (2017), which used a constrained optimization (CO) approach to match ride requests to vehicles. This approach is hard to scale and is myopic in assigning requests (it does not consider future rewards). The second baseline is the Neural Approximate Dynamic Programming (NeurADP) method from Shah, Lowalekar, and Varakantham (2020), which used a (centralized) DQN algorithm to learn a value function for effective mapping of requests. This approach assumes all agents are homogenous (i.e., having the same capacity and preferences), which is impractical. To keep comparisons fair, we consider a decentralized version of NeurADP as our baseline. Finally, we implement DMFG-QL for this problem where the mean-field corresponds to the distribution of ride requests at every node in the environment.

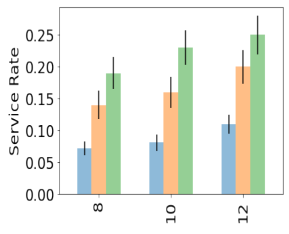
Similar to prior approaches (Lowalekar, Varakantham, and Jaillet 2019; Shah, Lowalekar, and Varakantham 2020), we use the service rate (total percentage of requests served) as the comparison metric. We train NeurADP and DMFG-QL using a set of eight consecutive days of training data and test all the performances in a previously unseen test set of six days. The test results are reported as an average (per day) of performances in the test set pertaining to three different hyperparameters. The first is the capacity of the vehicle ($c$) varied from 8 to 12, the second is the maximum allowed waiting time ($\tau$) varied from 520 seconds to 640 seconds, and the last is the number of vehicles ($N$), varied from 80 to 120. The results in Figs. 6(a-c) show that DMFG-QL outperforms the baselines in all our test cases. The mean field estimates in DMFG-QL help predict the distribution of ride requests in the environment, based on which agents can choose ride requests strategically. If agents choose orders that lead to destinations with a high percentage of requests, they will be able to serve more requests in the future. Thus, DMFG-QL outperforms



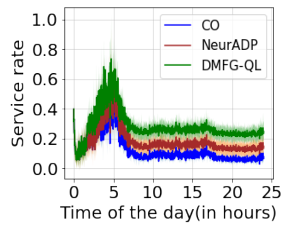
(a) # of Vehicles



(b) Maximum Pickup Delay



(c) Capacity



(d) Single Day Test

Figure 6: Results for the ride-sharing experiment. For (a), (b) and (c), we start with a prototypical configuration of $c$=10, $\tau$ = 580, and $N$ = 100, and then vary the different parameters. Figures (a), (b) and (c) share the same legend given in (a).

the NeurADP method (which does not maintain a mean field). We note that the service rate for all algorithms is low in this study (10% – 30%), since we are considering fewer vehicles compared to prior works (Shah, Lowalekar, and Varakantham 2020) due to the computational requirements of being decentralized. In practice our training is completely parallelizable and this is not a limitation of our approach. Also, from Fig. 6(c), an increase in the number of vehicles, increases the service rate. In Fig. 6(d) we plot the performance of the three algorithms for a single test day (24 hours — midnight to midnight). During certain times of the day (e.g., 5 am), the ride demand is low, and all approaches satisfy a large proportion of requests. However, during the other times of the day, when the demand is high, the DMFG-QL satisfies more requests than the baselines, showing its relative superiority.

## Conclusion

In this paper, we relaxed two strong assumptions in prior work on using mean field methods in RL. We introduced the DMFG framework, where agents are not assumed to have global information and are not homogeneous. All agents learn in a decentralized fashion, which contrasts with centralized procedures in prior work. Theoretically, we proved that the DMFG will have a suitable solution concept, DMFE. Also, we proved that a $Q$-learning based algorithm will find the DMFE. Further, we provided a principled method to address the 'chicken-and-egg' problem in MFRL, and demonstrated performances in a variety of environments (including RMP).

For future work, we would like to extend our theoretical analysis to the function approximation setting and analyze the convergence of policy gradient algorithms. Empirically, we could consider other real-world applications like autonomous driving and problems on demand and supply optimization.

## Acknowledgements

Resources used in preparing this research at the University of Waterloo were provided by the province of Ontario and the government of Canada through CIFAR, NRC, NSERC and companies sponsoring the Vector Institute. Part of this work has taken place in the Intelligent Robot Learning (IRL) Lab at the University of Alberta, which is supported in part by research grants from the Alberta Machine Intelligence Institute (Amii); a Canada CIFAR AI Chair, CIFAR; Compute Canada; and NSERC.

## References

Adlakha, S.; Johari, R.; and Weintraub, G. Y. 2015. Equilibria of dynamic games with many players: Existence, approximation, and market structure. *Journal of Economic Theory*, 156: 269–316.

Alonso-Mora, J.; Samaranayake, S.; Wallar, A.; Frazzoli, E.; and Rus, D. 2017. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of National Academy of Science USA*, 114(3): 462–467.

Anahtarci, B.; Kariksiz, C. D.; and Saldi, N. 2019. Value Iteration Algorithm for Mean-field Games. *arXiv preprint arXiv:1909.01758*.

Busoniu, L.; Babuska, R.; and De Schutter, B. 2006. Multi-agent reinforcement learning: A survey. In *2006 9th International Conference on Control, Automation, Robotics and Vision*, 1–6. IEEE.

Elie, R.; Pérolat, J.; Laurière, M.; Geist, M.; and Pietquin, O. 2020. On the Convergence of Model Free Learning in Mean Field Games. In *AAAI*, 7143–7150.

Elliott, R.; Li, X.; and Ni, Y.-H. 2013. Discrete time mean-field stochastic linear-quadratic optimal control problems. *Automatica*, 49(11): 3222–3233.

Gomes, D. A.; Mohr, J.; and Souza, R. R. 2010. Discrete time, finite state space mean field games. *Journal de mathématiques pures et appliquées*, 93(3): 308–328.

Guo, X.; Hu, A.; Xu, R.; and Zhang, J. 2019. Learning mean-field games. In *NeurIPS*, 4966–4976.

Gupta, J. K.; Egorov, M.; and Kochenderfer, M. 2017. Cooperative multi-agent control using deep reinforcement learning. In *AAMAS*, 66–83. Springer.

Hadikhanloo, S.; and Silva, F. J. 2019. Finite mean field games: fictitious play and convergence to a first order continuous mean field game. *Journal de Mathématiques Pures et Appliquées*, 132: 369–397.

He, H.; and Boyd-Graber, J. L. 2016. Opponent Modeling in Deep Reinforcement Learning. In *ICML*, volume 48, 1804–1813. JMLR.org.

Hernandez-Leal, P.; Kartal, B.; and Taylor, M. E. 2019. A survey and critique of multiagent deep reinforcement learning. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, 33(6): 750–797.

Hu, J.; and Wellman, M. P. 2003. Nash Q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov): 1039–1069.

Huang, M. 2010. Large-population LQG games involving a major player: the Nash certainty equivalence principle. *SIAM Journal on Control and Optimization*, 48(5): 3318–3353.

Huang, M.; Caines, P. E.; and Malhamé, R. P. 2003. Individual and mass behaviour in large population stochastic wireless power control problems: centralized and Nash equilibrium solutions. In *42nd IEEE International Conference on Decision and Control*. IEEE.

Huang, M.; Malhamé, R. P.; Caines, P. E.; et al. 2006. Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle. *Communications in Information & Systems*, 6(3): 221–252.

Konda, V. R.; and Tsitsiklis, J. N. 1999. Actor-Critic Algorithms. In *NeurIPS*. The MIT Press.

Lasry, J.-M.; and Lions, P.-L. 2007. Mean field games. *Japanese journal of mathematics*, 2(1): 229–260.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *ICLR*.

Lowalekar, M.; Varakantham, P.; and Jaillet, P. 2019. ZAC: A Zone Path Construction Approach for Effective Real-Time Ridesharing. In *ICAPS*, 528–538. AAAI Press.

Neumann, J. v. 1928. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1): 295–320.

NYYellowTaxi. 2016. New York yellow taxi dataset. http://www.nyc.gov/html/tlc/html/about/triprecorddata.shtml. [Online; accessed 19-June-2021].

Perrin, S.; Laurière, M.; Pérolat, J.; Geist, M.; Élie, R.; and Pietquin, O. 2021. Mean Field Games Flock! The Reinforcement Learning Way. In *IJCAI*, 356–362.

Saldi, N.; Basar, T.; and Raginsky, M. 2018. Discrete-time risk-sensitive mean-field games. *arXiv preprint arXiv:1808.03929*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. In *arXiv preprint arXiv:1707.06347*.

Shah, S.; Lowalekar, M.; and Varakantham, P. 2020. Neural Approximate Dynamic Programming for On-Demand Ride-Pooling. In *AAAI*, 507–515. AAAI Press.

Shoham, Y.; Powers, R.; and Grenager, T. 2003. Multi-agent reinforcement learning: a critical survey. Technical report, Technical report, Stanford University.

Shukla, S.; Balasubramanian, S.; and Pavlović, M. 2016. A generalized Banach fixed point theorem. *Bulletin of the Malaysian Mathematical Sciences Society*, 39(4): 1529–1539.

Stanley, H. E. 1971. Phase transitions and critical phenomena. Clarendon.

Subramanian, J.; and Mahajan, A. 2019. Reinforcement Learning in Stationary Mean-field Games. In *AAMAS*, 251–259. IFAAMAS.

Subramanian, S. G. 2021. Decentralized Mean Field Games. https://github.com/Sriram94/DMFG.

Subramanian, S. G.; Poupart, P.; Taylor, M. E.; and Hegde, N. 2020. Multi Type Mean Field Reinforcement Learning. In *AAMAS*. IFAAMAS.

Subramanian, S. G.; Taylor, M. E.; Crowley, M.; and Poupart, P. 2021. Partially Observable Mean Field Reinforcement Learning. In *AAMAS*. ACM.

Sutton, R. S.; McAllester, D. A.; Singh, S. P.; and Mansour, Y. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *NIPS*, 1057–1063. The MIT Press.

Tan, M. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *ICML*.

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

Więcek, P. 2020. Discrete-time ergodic mean-field games with average reward on compact spaces. *Dynamic Games and Applications*, 10(1): 222–256.

Więcek, P.; and Altman, E. 2015. Stationary anonymous sequential games with undiscounted rewards. *Journal of optimization theory and applications*, 166(2): 686–710.

Yang, Y.; Luo, R.; Li, M.; Zhou, M.; Zhang, W.; and Wang, J. 2018. Mean Field Multi-Agent Reinforcement Learning. In *ICML*. PMLR.

Yang, Y.; and Wang, J. 2020. An Overview of Multi-Agent Reinforcement Learning from Game Theoretical Perspective. *CoRR*, abs/2011.00583.

Zheng, L.; Yang, J.; Cai, H.; Zhou, M.; Zhang, W.; Wang, J.; and Yu, Y. 2018. MAgent: A many-agent reinforcement learning platform for artificial collective intelligence. In *AAAI*.